



@server

iSeries

Control Language (CL)





@server

iSeries

Control Language (CL)

Contents

Control language (CL)	1
What's new for V5R2	1
Print these topics	2
CL command finder	3
Alphabetic listing of commands	3
CL concepts and reference	44
Commands	45
Command names	45
Commands operating on OS/400 objects	45
Commands operating on multiple objects	46
Command description format	46
Syntax diagram format	47
Command parts	57
OS/400 objects	137
Commonly used parameters: Expanded descriptions	144
Database and device files used by CL commands	173
Printing command descriptions on the server	199
Related Information for CL.	199

Control language (CL)

Control language (CL) allows system programmers and system administrators to write programs using operating system commands and other IBM-supplied commands.

“What’s new for V5R2”

Select this link to find out about major CL changes for V5R2.

“Print these topics” on page 2

Select this link to print groups of CL commands or concepts.

“CL command finder” on page 3

Select this link to access the CL command finder, a tool that allows you to search for CL commands by name or description. You can even display an alphabetic list of all commands, all new commands, or all changed commands.

“Alphabetic listing of commands” on page 3

Select this link to display an alphabetic list of all commands by description.

“CL concepts and reference” on page 44

This topic describes the underlying concepts that you need to understand to effectively work with CL commands.

“Related Information for CL” on page 199

This topic provides a list of manuals and other information of interest.

What’s new for V5R2

The following lists information that is new to the Control Language (CL) topic in the Information Center for V5R2. To print CL information, including command descriptions, see “Print these topics” on page 2.

Following is a list of new object types and new system values.

New Object Types

The following object type is new in the OS/400 licensed program for Version 5 Release 2 Modification 0 (V5R2M0):

- *IMGCLG (Image catalog)

New and Changed Commands

Select the CL command finder link in the Information Center left navigation bar to access the CL command finder, a tool that allows you to search for CL commands by name or description. You can display an alphabetic list of commands, all new commands, or all changed commands.

The following products’ commands have been added to the Information Center for Version 5 Release 2 Modification 0 (V5R2M0):

- System Manager (product ID 5722SM1)
- Managed System Services (product ID 5722MG1)

To find other information about what’s new or changed this release, see the Memo to Users .

How to see what's new or changed:

To help you see where technical changes have been made to the CL information, this information uses:

- The



image to mark where new or changed information begins.

- The



image to mark where new or changed information ends.

In addition to the information about new or enhanced function listed above, the CL topic provides the following enhancements:

- A new CL command finder was added to make it easy to locate CL commands by description or by name.
- CL concepts and reference information was restructured for easier navigation.

To find other information about what's new or changed this release, see the Memo to Users .

Print these topics

The following links provide the PDF versions of the basic CL information, as well as the CL commands. (For information about printing command information from your server, see "Printing command descriptions on the server" on page 199).

The PDF versions of the commands **do not** contain syntax diagrams, but they can be seen and printed from the HTML. To view or download a PDF, select one of the following links:

CL Topic	Page Count/ KB
Control Language (basic CL information)	200/634
Commands ADDACC - ADDPRBSLTE	322/804
Commands ADDPRDCRQA - CHGCLURCY	244/628
Commands CHGCRG - CHGEWLM	316/756
Commands CHGFTR - CHGNODGRPA	356/885
Commands CHGOBJAUD - CHGTCPDMN	338/820
Commands CHGTCPHTE - CPYFRMIMPF	210/556
Commands CPYFRMPCD - CRTDEVSNUF	356/867
Commands CRTDEVTAP - CRTPTFPKG	422/1061
Commands CRTQMFORM - DLTJRN	184/510
Commands DLTJRNRCV - DSPAUTUSR	124/383
Commands DSPBCKUPL - DSPLANSTS	154/433
Commands DSPLOG - DSPUSRPMN	151/422
Commands DSPUSRPTI - ENDMGDSYS	124/344
Commands ENDMGRSRV - MRGMSGF	138/383
Commands MRGTCPTH - QRYTIEF	310/794
Commands RVDST - RMVLANADPT	156/459
Commands RMVLANADPI - RRTJOB	108/322


Commands RST - RUNRMTCMD	262/659
Commands RUNSQLSTM - STRCLUNOD	312/789
Commands STRCRG - STRUSF	154/425
Commands SBMCRQ - WRKDEVD	188/511
Commands WRKDEVTBL - WRKLNK	100/315
Commands WRKOBJLCK - WRKTAPCTG	104/313
Commands WRKTCPSTS - WRKWTR	10/70

Saving PDF files

To save a PDF on your workstation for viewing or printing:

1. Right-click the PDF in your browser (right-click the link above).
2. Click **Save Target As...**
3. Navigate to the directory in which you would like to save the PDF.
4. Click **Save**.

Downloading Adobe Acrobat Reader

If you need Adobe Acrobat Reader to view or print these PDFs, you can download a copy from the Adobe Web site (www.adobe.com/products/acrobat/readstep.html)  .

CL command finder

Use the CL command finder to find information about control language commands. You can search for CL commands or show a list of all the CL commands.

Alphabetic listing of commands

CL command finder: Select this link to access the CL command finder, a tool that allows you to search for CL commands by name or description. You can even display an alphabetic list of all commands, all new commands, or all changed commands.

A B C D E F G H I L M N O P Q R S T U
V W

A

- Add Access Code (ADDACC) command, ADDACC syntax diagram
- Add Alert Action Entry (ADDALRACNE) command, ADDALRACNE syntax diagram
- Add Alert Description (ADDALRD) command, ADDALRD syntax diagram
- Add Alert Selection Entry (ADDALRSLTE) command, ADDALRSLTE syntax diagram
- Add Authorization List Entry (ADDAUTLE) command, ADDAUTLE syntax diagram
- Add Autostart Job Entry (ADDAJE) command, ADDAJE syntax diagram
- Add Binding Directory Entry (ADDBNDDIRE) command, ADDBNDDIRE syntax diagram
- Add Breakpoint (ADDBKP) command, ADDBKP syntax diagram
- Add Change Control Server Client (ADDCCSCLT) command, ADDCCSCLT syntax diagram
- Add Change Request Activity (ADDCRQA) command, ADDCRQA syntax diagram
- Add Cluster Node Entry (ADDCLUNODE) command, ADDCLUNODE syntax diagram
- Add Cluster Resource Group Device Entry (ADDCRGDEVE) command, ADDCRGDEVE syntax diagram

- Add Cluster Resource Group Node Entry (ADDCRGNODE) command, ADDCRGNODE syntax diagram
- Add Command Change Request Activity (ADDCMDCRQA) command, ADDCMDCRQA syntax diagram
- Add Communications Entry (ADDCMNE) command, ADDCMNE syntax diagram
- Add Community for SNMP (ADDCOMSNMP) command, ADDCOMSNMP syntax diagram
- Add Configuration List Entries (ADDCFGLE) command, ADDCFGLE syntax diagram
- Add Connection List Entry (ADDCNNLE) command, ADDCNNLE syntax diagram
- Add Data Definition (ADDDTADFN) command, ADDDTADFN syntax diagram
- Add Device Domain Entry (ADDDEVDMNE) command, ADDDEVDMNE syntax diagram
- Add Directory Entry (ADDDIRE) command, ADDDIRE syntax diagram
- Add Directory Shadow System (ADDDIRSHD) command, ADDDIRSHD syntax diagram
- Add Distribution Catalog Entry (ADDDSTCLGE) command, ADDDSTCLGE syntax diagram
- Add Distribution List Entry (ADDDSTLE) command, ADDDSTLE syntax diagram
- Add Distribution Queue (ADDDSTQ) command, ADDDSTQ syntax diagram
- Add Distribution Route (ADDDSTRTE) command, ADDDSTRTE syntax diagram
- Add Distribution Secondary System Name (ADDDSTSYSN) command, ADDDSTSYSN syntax diagram
- Add Document Library Object Authority (ADDDLOAUT) command, ADDDLOAUT syntax diagram
- Add Emulation Configuration Entry (ADDEMLCFGE) command, ADDEMLCFGE syntax diagram
- Add Environment Variable (ADDENVVAR) command, ADDENVVAR syntax diagram
- Add Exit Program (ADDEXITPGM) command, ADDEXITPGM syntax diagram
- Add Extended Wireless Controller Bar Code Entry (ADDEWCBCDE) command, ADDEWCBCDE syntax diagram
- Add Extended Wireless Controller Member (ADDEWCM) command, ADDEWCM syntax diagram
- Add Extended Wireless Controller PTC Entry (ADDEWCPTCE) command, ADDEWCPTCE syntax diagram
- Add Extended Wireless Line Member (ADDEWLM) command, ADDEWLM syntax diagram
- Add Font Table Entry (ADDFNTTBLE) command, ADDFNTTBLE syntax diagram
- Add Host Database to DataLink File Manager (ADDHDBDLFM) command, ADDHDBDLFM syntax diagram
- Add Image Catalog Entry (ADDIMGCLGE) command, ADDIMGCLGE syntax diagram
- Add Intersystem Communications Function Program Device Entry (ADDICFDEVE) command, ADDICFDEVE syntax diagram
- Add IP over SNA Interface (ADDIPSIFC) command, ADDIPSIFC syntax diagram
- Add IP over SNA location entry (ADDIPSLOC) command, ADDIPSLOC syntax diagram
- Add IP over SNA Route (ADDIPSRTE) command, ADDIPSRTE syntax diagram
- Add Job Queue Entry (ADDJOBQE) command, ADDJOBQE syntax diagram
- Add Job Schedule Entry (ADDJOBSCDE) command, ADDJOBSCDE syntax diagram
- Add Job Using Job Scheduler (ADDJOBJS) command, ADDJOBJS syntax diagram
- Add Library List Entry (ADDLIBLE) command, ADDLIBLE syntax diagram
- Add License CRQ Activity (ADDLICCRQA) command, ADDLICCRQA syntax diagram
- Add License Key Information (ADDLICKEY) command, ADDLICKEY syntax diagram
- Add Link (ADDLNK) command, ADDLNK syntax diagram
- Add Local Area Network Adapter Information (ADDLANADPI) command, ADDLANADPI syntax diagram
- Add Logical File Member (ADDLFM) command, ADDLFM syntax diagram
- Add Media Information to BRM (ADDMEDIBRM) command, ADDMEDIBRM syntax diagram
- Add Media Library Media to BRM (ADDMLMBRM) command, ADDMLMBRM syntax diagram
- Add Media to BRM (ADDMEDBRM) command, ADDMEDBRM syntax diagram

- Add Message Description (ADDMSGD) command, ADDMSGD syntax diagram
- Add Mounted File System (ADDMFS) command, ADDMFS syntax diagram
- Add Mounted File System (MOUNT) command, MOUNT syntax diagram
- Add NetWare Authentication Entry (ADDNTWAUTE) command, ADDNTWAUTE syntax diagram
- Add Network Job Entry (ADDNETJOB) command, ADDNETJOB syntax diagram
- Add Network Server Storage Link (ADDNWSSTGL) command, ADDNWSSTGL syntax diagram
- Add Network Table Entry (ADDNETTBLE) command, ADDNETTBLE syntax diagram
- Add Nickname (ADDNCK) command, ADDNCK syntax diagram
- Add Node List Entry (ADDNODLE) command, ADDNODLE syntax diagram
- Add Object CRQ Activity (ADDOBJCRQA) command, ADDOBJCRQA syntax diagram
- Add Optical Cartridge (ADDOPTCTG) command, ADDOPTCTG syntax diagram
- Add Optical Server (ADDOPTSVR) command, ADDOPTSVR syntax diagram
- Add Performance Explorer Definition (ADDPEXDFN) command, ADDPEXDFN syntax diagram
- Add Performance Explorer Filter (ADDPEXFTR) command, ADDPEXFTR syntax diagram
- Add Physical File Constraint (ADDPFCST) command, ADDPFCST syntax diagram
- Add Physical File Member (ADDPFM) command, ADDPFM syntax diagram
- Add Physical File Trigger (ADDPFTRG) command, ADDPFTRG syntax diagram
- Add Point-to-Point TCP/IP Profile (ADDTCPPTP) command, ADDTCPPTP syntax diagram
- Add Prestart Job Entry (ADDPJE) command, ADDPJE syntax diagram
- Add Problem Action Entry (ADDPBACNE) command, ADDPBACNE syntax diagram
- Add Problem Selection Entry (ADDPBLSLE) command, ADDPBLSLE syntax diagram
- Add Product Change Request Activity (ADDPDCRQA) command, ADDPDCRQA syntax diagram
- Add Product License Information (ADDPDLICI) command, ADDPDLICI syntax diagram
- Add Program (ADDPGM) command, ADDPGM syntax diagram
- Add Program Temporary Fix Change Request Activity (ADDPTFCRQA) command, ADDPTFCRQA syntax diagram
- Add Protocol Table Entry (ADDPCLTBLE) command, ADDPCLTBLE syntax diagram
- Add Relational Database Directory Entry (ADDRDBDIRE) command, ADDRDBDIRE syntax diagram
- Add Remote Definition (ADDRMTDFN) command, ADDRMTDFN syntax diagram
- Add Remote Journal (ADDRMTJRN) command, ADDRMTJRN syntax diagram
- Add Reply List Entry (ADDRPYLE) command, ADDRPYLE syntax diagram
- Add Resource CRQ Activity (ADDRSCCRQA) command, ADDRSCCRQA syntax diagram
- Add REXX Buffer (ADDREXBUF) command, ADDREXBUF syntax diagram
- Add Routing Entry (ADDRTGE) command, ADDRTGE syntax diagram
- Add Search Index Entry (ADDSCHIDX) command, ADDSCHIDX syntax diagram
- Add Server Authentication Entry (ADDSVRAUTE) command, ADDSVRAUTE syntax diagram
- Add Service Table Entry (ADDSRVTBLE) command, ADDSRVTBLE syntax diagram
- Add Sphere of Control Entry (ADDSOCE) command, ADDSOCE syntax diagram
- Add Tape Cartridge (ADDTAPCTG) command, ADDTAPCTG syntax diagram
- Add TCP/IP Host Table Entry (ADDTCPHTE) command, ADDTCPHTE syntax diagram
- Add TCP/IP Interface (ADDTCPIFC) command, ADDTCPIFC syntax diagram
- Add TCP/IP Port Restriction (ADDTCPPORT) command, ADDTCPPORT syntax diagram
- Add TCP/IP Remote System Information (ADDTCPRSI) command, ADDTCPRSI syntax diagram
- Add TCP/IP Route (ADDTCPRTE) command, ADDTCPRTE syntax diagram
- Add Trace (ADDTRC) command, ADDTRC syntax diagram

- Add Trace Filter (ADDTRCFTR) command, ADDTRCFTR syntax diagram
- Add Ultimedia System Facilities Connection Entry (ADDUSFCNNE) command, ADDUSFCNNE syntax diagram
- Add Ultimedia System Facilities Device Entry (ADDUSFDEVE) command, ADDUSFDEVE syntax diagram
- Add Ultimedia System Facilities Server Entry (ADDUSFSVRE) command, ADDUSFSVRE syntax diagram
- Add Work Station Entry (ADDWSE) command, ADDWSE syntax diagram
- Allocate Object (ALCOBJ) command, ALCOBJ syntax diagram
- Analyze Access Group (ANZACCGRP) command, ANZACCGRP syntax diagram
- Analyze Database File (ANZDBF) command, ANZDBF syntax diagram
- Analyze Database File Keys (ANZDBFKEY) command, ANZDBFKEY syntax diagram
- Analyze Default Passwords (ANZDFTPWD) command, ANZDFTPWD syntax diagram
- Analyze Java Program (ANZJVAPGM) command, ANZJVAPGM syntax diagram
- Analyze Java Virtual Machine (ANZJVM) command, ANZJVM syntax diagram
- Analyze Libraries Using BRM (ANZLIBBRM) command, ANZLIBBRM syntax diagram
- Analyze Performance Data (ANZPFRDTA) command, ANZPFRDTA syntax diagram
- Analyze Problem (ANZPRB) command, ANZPRB syntax diagram
- Analyze Profile Activity (ANZPRFACT) command, ANZPRFACT syntax diagram
- Analyze Program (ANZPGM) command, ANZPGM syntax diagram
- Analyze Query (ANZQRY) command, ANZQRY syntax diagram
- Analyze User Objects (ANZUSROBJ) command, ANZUSROBJ syntax diagram
- Answer Line (ANSLIN) command, ANSLIN syntax diagram
- Answer Questions (ANSQST) command, ANSQST syntax diagram
- Apply Journalized Changes (APYJRNCHG) command, APYJRNCHG syntax diagram
- Apply Program Temporary Fix (APYPTF) command, APYPTF syntax diagram
- Apply Remote Program Temporary Fix (APYRMTPTF) command, APYRMTPTF syntax diagram
- Ask Question (ASKQST) command, ASKQST syntax diagram

Back to the top

B

- Batch Job (BCHJOB) command, BCHJOB syntax diagram

Back to the top

C

- Call Bound Procedure (CALLPRC) command, CALLPRC syntax diagram
- Call Program (CALL) command, CALL syntax diagram
- Change Accounting Code (CHGACGCDE) command, CHGACGCDE syntax diagram
- Change Activation Schedule Entry (CHGACTSCDE) command, CHGACTSCDE syntax diagram
- Change Active Profile List (CHGACTPRFL) command, CHGACTPRFL syntax diagram
- Change Alert Action Entry (CHGALRACNE) command, CHGALRACNE syntax diagram
- Change Alert Description (CHGALRD) command, CHGALRD syntax diagram
- Change Alert Selection Entry (CHGALRSLTE) command, CHGALRSLTE syntax diagram
- Change Alert Table (CHGALRTBL) command, CHGALRTBL syntax diagram
- Change ASP Attribute (CHGASPA) command, CHGASPA syntax diagram

- Change Attribute (CHGATR) command, CHGATR syntax diagram
- Change Auditing Value (CHGAUD) command, CHGAUD syntax diagram
- Change Authority (CHGAUT) command, CHGAUT syntax diagram
- Change Authorization List Entry (CHGAUTLE) command, CHGAUTLE syntax diagram
- Change Autostart Job Entry (CHGAJE) command, CHGAJE syntax diagram
- Change Backup Options (CHGBCKUP) command, CHGBCKUP syntax diagram
- Change BOOTP Attributes (CHGBPA) command, CHGBPA syntax diagram
- Change Change Control Server Attributes (CHGCCSA) command, CHGCCSA syntax diagram
- Change Change Request Activity (CHGCRQA) command, CHGCRQA syntax diagram
- Change Change Request Description (CHGCRQD) command, CHGCRQD syntax diagram
- Change Class (CHGCLS) command, CHGCLS syntax diagram
- Change Class-of-Service Description (CHGCOSD) command, CHGCOSD syntax diagram
- Change Cleanup (CHGCLNUP) command, CHGCLNUP syntax diagram
- Change Cluster Configuration Tuning (CHGCLUCFG) command, CHGCLUCFG syntax diagram
- Change Cluster Node Entry (CHGCLUNODE) command, CHGCLUNODE syntax diagram
- Change Cluster Recovery (CHGCLURCY) command, CHGCLURCY syntax diagram
- Change Cluster Resource Group (CHGCRG) command, CHGCRG syntax diagram
- Change Cluster Resource Group Device Entry (CHGCRGDEVE) command, CHGCRGDEVE syntax diagram
- Change Cluster Resource Group Primary (CHGCRGPRI) command, CHGCRGPRI syntax diagram
- Change Cluster Version (CHGCLUVER) command, CHGCLUVER syntax diagram
- Change Coded Font (CHGCDEFNT) command, CHGCDEFNT syntax diagram
- Change Command (CHGCMD) command, CHGCMD syntax diagram
- Change Command Change Request Activity (CHGCMDCRQA) command, CHGCMDCRQA syntax diagram
- Change Command Default (CHGCMDDFT) command, CHGCMDDFT syntax diagram
- Change Communications Entry (CHGCMNE) command, CHGCMNE syntax diagram
- Change Communications Side Information (CHGCSI) command, CHGCSI syntax diagram
- Change Community for SNMP (CHGCOMSNMP) command, CHGCOMSNMP syntax diagram
- Change Configuration List (CHGCFGL) command, CHGCFGL syntax diagram
- Change Configuration List Entry (CHGCFGLE) command, CHGCFGLE syntax diagram
- Change Connection List (CHGCNNL) command, CHGCNNL syntax diagram
- Change Connection List Entry (CHGCNNLE) command, CHGCNNLE syntax diagram
- Change Controller Description (APPC) (CHGCTLAPPC) command, CHGCTLAPPC syntax diagram
- Change Controller Description (Async) (CHGCTLASC) command, CHGCTLASC syntax diagram
- Change Controller Description (BSC) (CHGTLBSC) command, CHGTLBSC syntax diagram
- Change Controller Description (Finance) (CHGCTLFNC) command, CHGCTLFNC syntax diagram
- Change Controller Description (Local Work Station) (CHGCTLLWS) command, CHGCTLLWS syntax diagram
- Change Controller Description (Network) (CHGCTLNET) command, CHGCTLNET syntax diagram
- Change Controller Description (Remote Work Station) (CHGCTLRWS) command, CHGCTLRWS syntax diagram
- Change Controller Description (Retail) (CHGCTLRTL) command, CHGCTLRTL syntax diagram
- Change Controller Description (SNA Host) (CHGCTLHOST) command, CHGCTLHOST syntax diagram
- Change Controller Description (TAPE) (CHGCTLTAP) command, CHGCTLTAP syntax diagram

- Change Controller Description (Virtual Work Station) (CHGCTLVWS) command, CHGCTLVWS syntax diagram
- Change Current Directory (CD) command, CD syntax diagram
- Change Current Directory (CHDIR) command, CHDIR syntax diagram
- Change Current Directory (CHGCURDIR) command, CHGCURDIR syntax diagram
- Change Current Library (CHGCURLIB) command, CHGCURLIB syntax diagram
- Change Data Area (CHGDTAARA) command, CHGDTAARA syntax diagram
- Change DDM TCP/IP Attributes (CHGDDMTCPA) command, CHGDDMTCPA syntax diagram
- Change Debug (CHGDBG) command, CHGDBG syntax diagram
- Change Dedicated Service Tools Password (CHGDSTPWD) command, CHGDSTPWD syntax diagram
- Change Device Description (APPC) (CHGDEVAPPC) command, CHGDEVAPPC syntax diagram
- Change Device Description (ASP) (CHGDEVASP) command, CHGDEVASP syntax diagram
- Change Device Description (Async) (CHGDEVASC) command, CHGDEVASC syntax diagram
- Change Device Description (BSC) (CHGDEVBSC) command, CHGDEVBSC syntax diagram
- Change Device Description (Crypto) (CHGDEVCRP) command, CHGDEVCRP syntax diagram
- Change Device Description (Diskette) (CHGDEVDKT) command, CHGDEVDKT syntax diagram
- Change Device Description (Display) (CHGDEVDSP) command, CHGDEVDSP syntax diagram
- Change Device Description (Finance) (CHGDEVFNC) command, CHGDEVFNC syntax diagram
- Change Device Description (Intrasystem) (CHGDEVINTR) command, CHGDEVINTR syntax diagram
- Change Device Description (Media Library) (CHGDEVMLB) command, CHGDEVMLB syntax diagram
- Change Device Description (Network) (CHGDEVNET) command, CHGDEVNET syntax diagram
- Change Device Description (Optical) (CHGDEVOPT) command, CHGDEVOPT syntax diagram
- Change Device Description (Printer) (CHGDEVPRT) command, CHGDEVPRT syntax diagram
- Change Device Description (Retail) (CHGDEVRTL) command, CHGDEVRTL syntax diagram
- Change Device Description (SNA Host) (CHGDEVHOST) command, CHGDEVHOST syntax diagram
- Change Device Description (SNA Pass-Through) (CHGDEVSNTPT) command, CHGDEVSNTPT syntax diagram
- Change Device Description (SNUF) (CHGDEVSNUF) command, CHGDEVSNUF syntax diagram
- Change Device Description (Tape) (CHGDEVTAP) command, CHGDEVTAP syntax diagram
- Change DHCP Attributes (CHGDHCPA) command, CHGDHCPA syntax diagram
- Change Directory Entry (CHGDIRE) command, CHGDIRE syntax diagram
- Change Directory Shadow System (CHGDIRSHD) command, CHGDIRSHD syntax diagram
- Change Diskette File (CHGDKTF) command, CHGDKTF syntax diagram
- Change Display File (CHGDSPF) command, CHGDSPF syntax diagram
- Change Distributed Data Management File (CHGDDMF) command, CHGDDMF syntax diagram
- Change Distribution Attributes (CHGDSTA) command, CHGDSTA syntax diagram
- Change Distribution Description (CHGDSTD) command, CHGDSTD syntax diagram
- Change Distribution List (CHGDSTL) command, CHGDSTL syntax diagram
- Change Distribution Queue (CHGDSTQ) command, CHGDSTQ syntax diagram
- Change Distribution Route (CHGDSTRTE) command, CHGDSTRTE syntax diagram
- Change Document Description (CHGDOCD) command, CHGDOCD syntax diagram
- Change Document Library Object Audit (CHGDLOAUD) command, CHGDLOAUD syntax diagram
- Change Document Library Object Authority (CHGDLOAUT) command, CHGDLOAUT syntax diagram
- Change Document Library Object Owner (CHGDLOOWN) command, CHGDLOOWN syntax diagram

- Change Document Library Object Primary Group (CHGDLOPGP) command, CHGDLOPGP syntax diagram
- Change Emulation Configuration Entry (CHGEMLCFGE) command, CHGEMLCFGE syntax diagram
- Change Environment Variable (CHGENVVAR) command, CHGENVVAR syntax diagram
- Change Expiration Schedule Entry (CHGEXPSCDE) command, CHGEXPSCDE syntax diagram
- Change Extended Wireless Controller Bar Code Entry (CHGEWCBCDE) command, CHGEWCBCDE syntax diagram
- Change Extended Wireless Controller Member (CHGEWCM) command, CHGEWCM syntax diagram
- Change Extended Wireless Controller PTC Entry (CHGEWCPTCE) command, CHGEWCPTCE syntax diagram
- Change Extended Wireless Line Member (CHGEWLM) command, CHGEWLM syntax diagram
- Change Filter (CHGFTR) command, CHGFTR syntax diagram
- Change Font Resource (CHGFNTRSC) command, CHGFNTRSC syntax diagram
- Change Font Table Entry (CHGFNTTBLE) command, CHGFNTTBLE syntax diagram
- Change Functional Area (CHGFCNARA) command, CHGFCNARA syntax diagram
- Change Graph Format (CHGGPHFMT) command, CHGGPHFMT syntax diagram
- Change Graph Package (CHGGPHPKG) command, CHGGPHPKG syntax diagram
- Change Group Attributes (CHGGRPA) command, CHGGRPA syntax diagram
- Change High-Level Language Pointer (CHGHLLPTR) command, CHGHLLPTR syntax diagram
- Change HTTP Attributes (CHGHTTPA) command
- Change ICF Program Device Entry (CHGICFDEVE) command, CHGICFDEVE syntax diagram
- Change Image Catalog (CHGIMGCLG) command, CHGIMGCLG syntax diagram
- Change Image Catalog Entry (CHGIMGCLGE) command, CHGIMGCLGE syntax diagram
- Change Intersystem Communications Function File (CHGICFF) command, CHGICFF syntax diagram
- Change IP over SNA Interface (CHGIPSIFC) command, CHGIPSIFC syntax diagram
- Change IP over SNA Location (CHGIPSLOC) command, CHGIPSLOC syntax diagram
- Change IP over SNA Type of Service (CHGIPSTOS) command, CHGIPSTOS syntax diagram
- Change IPL Attributes (CHGIPLA) command, CHGIPLA syntax diagram
- Change Java Program (CHGJVAPGM) command, CHGJVAPGM syntax diagram
- Change Job (CHGJOB) command, CHGJOB syntax diagram
- Change Job Authority Using Job Scheduler (CHGAUTJS) command, CHGAUTJS syntax diagram
- Change Job Description (CHGJOBBD) command, CHGJOBBD syntax diagram
- Change Job Media Library Attributes (CHGJOBMLBA) command, CHGJOBMLBA syntax diagram
- Change Job Queue Entry (CHGJOBQE) command, CHGJOBQE syntax diagram
- Change Job Schedule Entry (CHGJOBSCDE) command, CHGJOBSCDE syntax diagram
- Change Job Scheduler (CHGSCDBRM) command, CHGSCDBRM syntax diagram
- Change Job Type (CHGJOBTYP) command, CHGJOBTYP syntax diagram
- Change Job Using Job Scheduler (CHGJOBJS) command, CHGJOBJS syntax diagram
- Change Journal (CHGJRN) command, CHGJRN syntax diagram
- Change Keyboard Map (CHGKBDMAP) command, CHGKBDMAP syntax diagram
- Change LAN Adapter Information (CHGLANADPI) command, CHGLANADPI syntax diagram
- Change Library (CHGLIB) command, CHGLIB syntax diagram
- Change Library List (CHGLIBL) command, CHGLIBL syntax diagram
- Change License CRQ Activity (CHGLICCRQA) command, CHGLICCRQA syntax diagram
- Change License Information (CHGLICINF) command, CHGLICINF syntax diagram

- Change Line Description (Async) (CHGLINASC) command, CHGLINASC syntax diagram
- Change Line Description (BSC) (CHGLINBSC) command, CHGLINBSC syntax diagram
- Change Line Description (DDI Network) (CHGLINDDI) command, CHGLINDDI syntax diagram
- Change Line Description (Ethernet) (CHGLINETH) command, CHGLINETH syntax diagram
- Change Line Description (Fax) (CHGLINFAX) command, CHGLINFAX syntax diagram
- Change Line Description (Frame Relay Network) (CHGLINFR) command, CHGLINFR syntax diagram
- Change Line Description (IDLC) (CHGLINIDLC) command, CHGLINIDLC syntax diagram
- Change Line Description (Network) (CHGLINNET) command, CHGLINNET syntax diagram
- Change Line Description (PPP) (CHGLINPPP) command, CHGLINPPP syntax diagram
- Change Line Description (SDLC) (CHGLINSDLC) command, CHGLINSDLC syntax diagram
- Change Line Description (TDLC) (CHGLINTDLC) command, CHGLINTDLC syntax diagram
- Change Line Description (Token-Ring Network) (CHGLINTRN) command, CHGLINTRN syntax diagram
- Change Line Description (Wireless) (CHGLINWLS) command, CHGLINWLS syntax diagram
- Change Line Description (X.25) (CHGLINX25) command, CHGLINX25 syntax diagram
- Change Logical File (CHGLF) command, CHGLF syntax diagram
- Change Logical File Member (CHGLFM) command, CHGLFM syntax diagram
- Change Managed System Attributes (CHGMGDSYSA) command, CHGMGDSYSA syntax diagram
- Change Manager Services Attributes (CHGMGRSRVA) command, CHGMGRSRVA syntax diagram
- Change Media Using BRM (CHGMEDBRM) command, CHGMEDBRM syntax diagram
- Change Menu (CHGMNU) command, CHGMNU syntax diagram
- Change Message Description (CHGMSGD) command, CHGMSGD syntax diagram
- Change Message File (CHGMSGF) command, CHGMSGF syntax diagram
- Change Message Queue (CHGMSGQ) command, CHGMSGQ syntax diagram
- Change Mode Description (CHGMODD) command, CHGMODD syntax diagram
- Change Module (CHGMOD) command, CHGMOD syntax diagram
- Change NetBIOS Description (CHGNTBD) command, CHGNTBD syntax diagram
- Change NetWare Authentication Entry (CHGNTWAUTE) command, CHGNTWAUTE syntax diagram
- Change NetWare Volume (CHGNTWVOL) command, CHGNTWVOL syntax diagram
- Change Network Attributes (CHGNETA) command, CHGNETA syntax diagram
- Change Network File System Export (CHGNFSEXP) command, CHGNFSEXP syntax diagram
- Change Network File System Export (EXPORTFS) command, EXPORTFS syntax diagram
- Change Network Interface (ATM Network) (CHGNWIATM) command, CHGNWIATM syntax diagram
- Change Network Interface (Frame Relay Network) (CHGNWIFR) command, CHGNWIFR syntax diagram
- Change Network Interface Description for ISDN (CHGNWIISDN) command, CHGNWIISDN syntax diagram
- Change Network Job Entry (CHGNETJOB) command, CHGNETJOB syntax diagram
- Change Network Server Attributes (CHGNWSA) command, CHGNWSA syntax diagram
- Change Network Server Description (CHGNWSD) command, CHGNWSD syntax diagram
- Change Network Server User Attributes (CHGNWSUSRA) command, CHGNWSUSRA syntax diagram
- Change Network Time Protocol Attributes (CHGNTPA) command, CHGNTPA syntax diagram
- Change Nickname (CHGNCK) command, CHGNCK syntax diagram
- Change Node Group Attributes (CHGNODGRPA) command, CHGNODGRPA syntax diagram
- Change Object Auditing (CHGOBJAUD) command, CHGOBJAUD syntax diagram
- Change Object Change Request Activity (CHGOBJCRQA) command, CHGOBJCRQA syntax diagram

- Change Object Description (CHGOBJD) command, CHGOBJD syntax diagram
- Change Object Owner (CHGOBJOWN) command, CHGOBJOWN syntax diagram
- Change Object Primary Group (CHGOBJPGP) command, CHGOBJPGP syntax diagram
- Change Optical Attributes (CHGOPTA) command, CHGOPTA syntax diagram
- Change Optical Volume (CHGOPTVOL) command, CHGOPTVOL syntax diagram
- Change Output Queue (CHGOUTQ) command, CHGOUTQ syntax diagram
- Change Owner (CHGOWN) command, CHGOWN syntax diagram
- Change Pager Command Using Job Scheduler (CHGPGRJS) command, CHGPGRJS syntax diagram
- Change Parmeter Data (CHGDTAJS) command, CHGDTAJS syntax diagram
- Change Password (CHGPWD) command, CHGPWD syntax diagram
- Change Performance Explorer Definition (CHGPEXDFN) command, CHGPEXDFN syntax diagram
- Change Physcial File Trigger (CHGPFTRG) command, CHGPFTRG syntax diagram
- Change Physical File (CHGPF) command, CHGPF syntax diagram
- Change Physical File Constraint (CHGPF CST) command, CHGPF CST syntax diagram
- Change Physical File Member (CHGPFM) command, CHGPFM syntax diagram
- Change Pointer (CHGPTR) command, CHGPTR syntax diagram
- Change Power On/Off Schedule (CHGPWRSCD) command, CHGPWRSCD syntax diagram
- Change Power On/Off Schedule Entry (CHGPWRSCDE) command, CHGPWRSCDE syntax diagram
- Change Prestart Job (CHGPJ) command, CHGPJ syntax diagram
- Change Prestart Job Entry (CHGPJE) command, CHGPJE syntax diagram
- Change Prestart Job Entry (CHGPJE) command, CHGPJE syntax diagram
- Change Primary Group (CHGPGP) command, CHGPGP syntax diagram
- Change Print Descriptor Group Profile (CHGPDGPRF) command, CHGPDGPRF syntax diagram
- Change Print Services Facility Configuration (CHGPSFCFG) command, CHGPSFCFG syntax diagram
- Change Printer File (CHGPRTF) command, CHGPRTF syntax diagram
- Change Problem (CHGPRB) command, CHGPRB syntax diagram
- Change Problem Action Entry (CHGPRBACNE) command, CHGPRBACNE syntax diagram
- Change Problem Selection Entry (CHGPRBSLTE) command, CHGPRBSLTE syntax diagram
- Change Product Change Request Activity (CHGPRDCRQA) command, CHGPRDCRQA syntax diagram
- Change Product Object Description (CHGPRDOBJD) command, CHGPRDOBJD syntax diagram
- Change Profile (CHGPRF) command, CHGPRF syntax diagram
- Change Program (CHGPGM) command, CHGPGM syntax diagram
- Change Program Variable (CHGPGMVAR) command, CHGPGMVAR syntax diagram
- Change PTF Change Request Activity (CHGPTFCRQA) command, CHGPTFCRQA syntax diagram
- Change Query Attributes (CHGQRYA) command, CHGQRYA syntax diagram
- Change Question-and-Answer Database (CHGQSTDB) command, CHGQSTDB syntax diagram
- Change Recovery for Access Paths (CHGRCYAP) command, CHGRCYAP syntax diagram
- Change Relational Database Directory Entry (CHGRDBDIRE) command, CHGRDBDIRE syntax diagram
- Change Remote Definition (CHGRMTDFN) command, CHGRMTDFN syntax diagram
- Change Remote Journal (CHGRMTJRN) command, CHGRMTJRN syntax diagram
- Change Reply List Entry (CHGRPYLE) command, CHGRPYLE syntax diagram
- Change Resource Change Request Activity (CHGRSCCRQA) command, CHGRSCCRQA syntax diagram
- Change Routed Attributes (CHGRTDA) command, CHGRTDA syntax diagram

- Change Routing Entry (CHGRTGE) command, CHGRTGE syntax diagram
- Change RWS Controller Password (CHGRWSPWD) command, CHGRWSPWD syntax diagram
- Change Save File (CHGSAVF) command, CHGSAVF syntax diagram
- Change Search Index (CHGSCHIDX) command, CHGSCHIDX syntax diagram
- Change Security Attributes (CHGSECA) command, CHGSECA syntax diagram
- Change Security Auditing Values (CHGSECAUD) command, CHGSECAUD syntax diagram
- Change Server Authentication Entry (CHGSVRAUTE) command, CHGSVRAUTE syntax diagram
- Change Service Attributes (CHGSRVA) command, CHGSRVA syntax diagram
- Change Service Program (CHGSRVPGM) command, CHGSRVPGM syntax diagram
- Change Service Provider Attributes (CHGSRVPVDA) command, CHGSRVPVDA syntax diagram
- Change Session Maximum (CHGSSNMAX) command, CHGSSNMAX syntax diagram
- Change Shared Storage Pool (CHGSHRPOOL) command, CHGSHRPOOL syntax diagram
- Change SNMP Attributes (CHGSNMPA) command, CHGSNMPA syntax diagram
- Change Source Physical File (CHGSRCPF) command, CHGSRCPF syntax diagram
- Change Spooled File Attributes (CHGSPLFA) command, CHGSPLFA syntax diagram
- Change Subsystem Description (CHGSBSD) command, CHGSBSD syntax diagram
- Change System Directory Attributes (CHGSYSDIRA) command, CHGSYSDIRA syntax diagram
- Change System Job (CHGSYSJOB) command, CHGSYSJOB syntax diagram
- Change System Library List (CHGSYSLIBL) command, CHGSYSLIBL syntax diagram
- Change System Value (CHGSYSVAL) command, CHGSYSVAL syntax diagram
- Change System/36 (CHGS36) command, CHGS36 syntax diagram
- Change System/36 Attributes (CHGS36A) command, CHGS36A syntax diagram
- Change System/36 Message List (CHGS36MSGL) command, CHGS36MSGL syntax diagram
- Change System/36 Procedure Attributes (CHGS36PRCA) command, CHGS36PRCA syntax diagram
- Change System/36 Program Attributes (CHGS36PGMA) command, CHGS36PGMA syntax diagram
- Change System/36 Source Attributes (CHGS36SRCA) command, CHGS36SRCA syntax diagram
- Change Tape Cartridge (CHGTAPCTG) command, CHGTAPCTG syntax diagram
- Change Tape File (CHGTAPF) command, CHGTAPF syntax diagram
- Change TCP/IP Attributes (CHGTCPA) command, CHGTCPA syntax diagram
- Change TCP/IP Domain (CHGTCPDMN) command, CHGTCPDMN syntax diagram
- Change TCP/IP Host Table Entry (CHGTCPHTE) command, CHGTCPHTE syntax diagram
- Change TCP/IP Interface (CHGTCPIFC) command, CHGTCPIFC syntax diagram
- Change TCP/IP Route (CHGTCPRTE) command, CHGTCPRTE syntax diagram
- Change TFTP Server Attributes (CHGTFTPA) command, CHGTFTPA syntax diagram
- Change Ultimedia System Facilities Device Entry (CHGUSFDEVE) command, CHGUSFDEVE syntax diagram
- Change User Audit (CHGUSRAUD) command, CHGUSRAUD syntax diagram
- Change User Print Information (CHGUSRPRTI) command, CHGUSRPRTI syntax diagram
- Change User Profile (CHGUSRPRF) command, CHGUSRPRF syntax diagram
- Change User Trace Buffer (CHGUSRTRC) command, CHGUSRTRC syntax diagram
- Change Variable (CHGVAR) command, CHGVAR syntax diagram
- Change Work Station Entry (CHGWSE) command, CHGWSE syntax diagram
- Change Writer (CHGWTR) command, CHGWTR syntax diagram
- Check ASP Balance (CHKASPBAL) command, CHKASPBAL syntax diagram
- Check Communications Trace (CHKCMNTRC) command, CHKCMNTRC syntax diagram

- Check DBCS Font Table (CHKIGCTBL) command, CHKIGCTBL syntax diagram
- Check Diskette (CHKDKT) command, CHKDKT syntax diagram
- Check Document Library Object (CHKDLO) command, CHKDLO syntax diagram
- Check Expired Media for BRM (CHKEXPBRM) command, CHKEXPBRM syntax diagram
- Check In (CHKIN) command, CHKIN syntax diagram
- Check Object (CHKOBJ) command, CHKOBJ syntax diagram
- Check Object Integrity (CHKOBJITG) command, CHKOBJITG syntax diagram
- Check Out (CHKOUT) command, CHKOUT syntax diagram
- Check Password (CHKPWD) command, CHKPWD syntax diagram
- Check Product Option (CHKPRDOPT) command, CHKPRDOPT syntax diagram
- Check Record Locks (CHKRCDLCK) command, CHKRCDLCK syntax diagram
- Check Tape (CHKTAP) command, CHKTAP syntax diagram
- Clear Diskette (CLRDKT) command, CLRDKT syntax diagram
- Clear Job Queue (CLRJOBQ) command, CLRJOBQ syntax diagram
- Clear Library (CLRLIB) command, CLRLIB syntax diagram
- Clear Message Queue (CLRMSGQ) command, CLRMSGQ syntax diagram
- Clear Output Queue (CLROUTQ) command, CLROUTQ syntax diagram
- Clear Physical File Member (CLRPFM) command, CLRPFM syntax diagram
- Clear Pool (CLRPOOL) command, CLRPOOL syntax diagram
- Clear Save File (CLRSAVF) command, CLRSAVF syntax diagram
- Clear Server Security Data (CLRSVRSEC) command, CLRSVRSEC syntax diagram
- Clear Trace Data (CLRTRCDTA) command, CLRTRCDTA syntax diagram
- Close File (CLOF) command, CLOF syntax diagram
- Commit (COMMIT) command, COMMIT syntax diagram
- Compare Journal Images (CMPJRNIMG) command, CMPJRNIMG syntax diagram
- Compress Object (CPROBJ) command, CPROBJ syntax diagram
- Configure Device Media Library (CFGDEVMLB) command, CFGDEVMLB syntax diagram
- Configure Distribution Services (CFGDSTSRV) command, CFGDSTSRV syntax diagram
- Configure HTTP Search (CFGHTTPSCH) command
- Configure PM/400 (CFGPM400) command, CFGPM400 syntax diagram
- Configure Point-to-Point TCP/IP (CFGTCPPTP) command, CFGTCPPTP syntax diagram
- Configure System Security (CFGSYSSEC) command, CFGSYSSEC syntax diagram
- Configure TCP/IP (CFGTCP) command, CFGTCP syntax diagram
- Configure TCP/IP Applications (FGTCPAPP) command, CFGTCPAPP syntax diagram
- Configure TCP/IP BOOTP (FGTCPBP) command, CFGTCPBP syntax diagram
- Configure TCP/IP HTTP (FGTCPHTTP) command
- Configure TCP/IP Routed (FGTCPRTD) command, CFGTCPRTD syntax diagram
- Configure TCP/IP SNMP (FGTcpsnmp) command, CFGTcpsnmp syntax diagram
- Convert CL Source (CVTCLSRC) command, CVTCLSRC syntax diagram
- Convert Date (CVTDAT) command, CVTDAT syntax diagram
- Convert Directory (CVTDIR) command, CVTDIR syntax diagram
- Convert Document Library Services Name (CVTDLSNAM) command, CVTDLSNAM syntax diagram
- Convert Education (CVTEDU) command, CVTEDU syntax diagram
- Convert IP over SNA Interface (CVTIPSIFC) command, CVTIPSIFC syntax diagram
- Convert IP over SNA Location Entry (CVTIPSLOC) command, CVTIPSLOC syntax diagram

- Convert Optical Backup (CVTOPTBKU) command, CVTOPTBKU syntax diagram
- Convert Performance Data (CVTPFRDTA) command, CVTPFRDTA syntax diagram
- Convert Performance Thread Data (CVTPFRTHD) command, CVTPFRTHD syntax diagram
- Convert RPC Source (CVTRPCSRC) command, CVTRPCSRC syntax diagram
- Convert RPC Source (RPCGEN) command, RPCGEN syntax diagram
- Convert TCP/IP CL Source (CVTTCPCL) command, CVTTCPCL syntax diagram
- Copy (COPY) command, COPY syntax diagram
- Copy (CPY) command, CPY syntax diagram
- Copy Configuration List (CPYCFGL) command, CPYCFGL syntax diagram
- Copy DBCS Font Table (CPYIGCTBL) command, CPYIGCTBL syntax diagram
- Copy Distribution Repository Object (CPYDSTRPSO) command, CPYDSTRPSO syntax diagram
- Copy Document (CPYDOC) command, CPYDOC syntax diagram
- Copy File (CPYF) command, CPYF syntax diagram
- Copy From Directory (CPYFRMDIR) command, CPYFRMDIR syntax diagram
- Copy from Diskette (CPYFRMDKT) command, CPYFRMDKT syntax diagram
- Copy From Import File (CPYFRMIMPF) command, CPYFRMIMPF syntax diagram
- Copy from PC Document (CPYFRMPCD) command, CPYFRMPCD syntax diagram
- Copy From PCF File (CPYFRMPCFF) command, CPYFRMPCFF syntax diagram
- Copy From Query File (CPYFRMQRYF) command, CPYFRMQRYF syntax diagram
- Copy from Stream File (CPYFRMSTMF) command, CPYFRMSTMF syntax diagram
- Copy from Tape (CPYFRMTAP) command, CPYFRMTAP syntax diagram
- Copy Functional Area (CPYFCNARA) command, CPYFCNARA syntax diagram
- Copy Graph Format (CPYGPHFMT) command, CPYGPHFMT syntax diagram
- Copy Graph Package (CPYGPHPKG) command, CPYGPHPKG syntax diagram
- Copy Job Using Job Scheduler (CPYJOBJS) command, CPYJOBJS syntax diagram
- Copy Library (CPYLIB) command, CPYLIB syntax diagram
- Copy Media Information Using BRM (CPYMEDIBRM) command, CPYMEDIBRM syntax diagram
- Copy Optical (CPYOPT) command, CPYOPT syntax diagram
- Copy Performance Data (CPYPFRDTA) command, CPYPFRDTA syntax diagram
- Copy Program Temporary Fix (CPYPTF) command, CPYPTF syntax diagram
- Copy PTF Cover Letter (CPYPTFCVR) command, CPYPTFCVR syntax diagram
- Copy PTF Group (CPYPTFGRP) command, CPYPTFGRP syntax diagram
- Copy PTF to Save File (CPYPTFSAVF) command, CPYPTFSAVF syntax diagram
- Copy Source File (CPYSRCF) command, CPYSRCF syntax diagram
- Copy Spooled File (CPYSPLF) command, CPYSPLF syntax diagram
- Copy To Directory (CPYTODIR) command, CPYTODIR syntax diagram
- Copy to Diskette (CPYTODKT) command, CPYTODKT syntax diagram
- Copy To Import File (CPYTOIMPF) command, CPYTOIMPF syntax diagram
- Copy to PC Document (CPYTOPCD) command, CPYTOPCD syntax diagram
- Copy To PCF File (CPYTOPCFF) command, CPYTOPCFF syntax diagram
- Copy to Stream File (CPYTOSTMF) command, CPYTOSTMF syntax diagram
- Copy to Tape (CPYTOTAP) command, CPYTOTAP syntax diagram
- COPYRIGHT (COPYRIGHT) command, COPYRIGHT syntax diagram
- Create a Data Dictionary (CRTDTADCT) command, CRTDTADCT syntax diagram
- Create Alert Table (CRTALRTBL) command, CRTALRTBL syntax diagram

- Create Authority Holder (CRTAUTHLR) command, CRTAUTHLR syntax diagram
- Create Authorization List (CRTAUTL) command, CRTAUTL syntax diagram
- Create Binding Directory (CRTBNDDIR) command, CRTBNDDIR syntax diagram
- Create Bound Control Language Program (CRTBNDCL) command, CRTBNDCL syntax diagram
- Create Change Request Description (CRTCRQD) command, CRTCRQD syntax diagram
- Create Class (CRTCLS) command, CRTCLS syntax diagram
- Create Class-of-Service Description (CRTCOSD) command, CRTCOSD syntax diagram
- Create Cluster (CRTCLU) command, CRTCLU syntax diagram
- Create Cluster Resource Group (CRTCRG) command, CRTCRG syntax diagram
- Create Command (CRTCMD) command, CRTCMD syntax diagram
- Create Communications Side Information (CRTCSI) command, CRTCSI syntax diagram
- Create Configuration List (CRTCFGL) command, CRTCFGL syntax diagram
- Create Connection List (CRTCNNL) command, CRTCNNL syntax diagram
- Create Control Language Module (CRTCLMOD) command, CRTCLMOD syntax diagram
- Create Control Language Program (CRTCLPGM) command, CRTCLPGM syntax diagram
- Create Controller Description (APPC) (CRTCTLAPPC) command, CRTCTLAPPC syntax diagram
- Create Controller Description (Async) (CRTCTLASC) command, CRTCTLASC syntax diagram
- Create Controller Description (BSC) (CRTCTLBSC) command, CRTCTLBSC syntax diagram
- Create Controller Description (Finance) (CRTCTLFNC) command, CRTCTLFNC syntax diagram
- Create Controller Description (Local Work Station) (CRTCTLLWS) command, CRTCTLLWS syntax diagram
- Create Controller Description (Network) (CRTCTLNET) command, CRTCTLNET syntax diagram
- Create Controller Description (Remote Work Station) (CRTCTLRWS) command, CRTCTLRWS syntax diagram
- Create Controller Description (Retail) (CRTCTLRTL) command, CRTCTLRTL syntax diagram
- Create Controller Description (SNA Host) (CRTCTLHOST) command, CRTCTLHOST syntax diagram
- Create Controller Description (Tape) (CRTCTLTAP) command, CRTCTLTAP syntax diagram
- Create Controller Description (Virtual Work Station) (CRTCTLVWS) command, CRTCTLVWS syntax diagram
- Create Data Area (CRTDTAARA) command, CRTDTAARA syntax diagram
- Create Data Queue (CRTDTAQ) command, CRTDTAQ syntax diagram
- Create DBCS Conversion Dictionary (CRTIGCDCT) command, CRTIGCDCT syntax diagram
- Create Device Description (APPC) (CRTDEVAPPC) command, CRTDEVAPPC syntax diagram
- Create Device Description (ASP) (CRTDEVASP) command, CRTDEVASP syntax diagram
- Create Device Description (Async) (CRTDEVASC) command, CRTDEVASC syntax diagram
- Create Device Description (BSC) (CRTDEVBSC) command, CRTDEVBSC syntax diagram
- Create Device Description (Crypto) (CRTDEVCRP) command, CRTDEVCRP syntax diagram
- Create Device Description (Diskette) (CRTDEVDKT) command, CRTDEVDKT syntax diagram
- Create Device Description (Display) (CRTDEVDSP) command, CRTDEVDSP syntax diagram
- Create Device Description (Finance) (CRTDEVFNC) command, CRTDEVFNC syntax diagram
- Create Device Description (Intrasystem) (CRTDEVINTR) command, CRTDEVINTR syntax diagram
- Create Device Description (Media Library) (CRTDEVMLB) command, CRTDEVMLB syntax diagram
- Create Device Description (Network) (CRTDEVNET) command, CRTDEVNET syntax diagram
- Create Device Description (Optical) (CRTDEVOPT) command, CRTDEVOPT syntax diagram
- Create Device Description (Printer) (CRTDEVPRT) command, CRTDEVPRT syntax diagram

- Create Device Description (Retail) (CRTDEVRTL) command, CRTDEVRTL syntax diagram
- Create Device Description (SNA Host) (CRTDEVHOST) command, CRTDEVHOST syntax diagram
- Create Device Description (SNA Pass-Through) (CRTDEVSNTPT) command, CRTDEVSNTPT syntax diagram
- Create Device Description (SNUF) (CRTDEVSNUF) command, CRTDEVSNUF syntax diagram
- Create Device Description (Tape) (CRTDEVTAP) command, CRTDEVTAP syntax diagram
- Create Directory (CRTDIR) command, CRTDIR syntax diagram
- Create Directory (MD) command, MD syntax diagram
- Create Directory (MKDIR) command, MKDIR syntax diagram
- Create Diskette File (CRTDKTF) command, CRTDKTF syntax diagram
- Create Display File (CRTDSPF) command, CRTDSPF syntax diagram
- Create Distributed Data Management File (CRTDDMF) command, CRTDDMF syntax diagram
- Create Distribution List (CRTDSTL) command, CRTDSTL syntax diagram
- Create Document (CRTDOC) command, CRTDOC syntax diagram
- Create Duplicate Object (CRTDUPOBJ) command, CRTDUPOBJ syntax diagram
- Create Edit Description (CRTEDTD) command, CRTEDTD syntax diagram
- Create Filter (CRTFTR) command, CRTFTR syntax diagram
- Create Folder (CRTFLR) command, CRTFLR syntax diagram
- Create Font Resources (CRTFNTRSC) command, CRTFNTRSC syntax diagram
- Create Font Table (CRTFNNTBL) command, CRTFNNTBL syntax diagram
- Create Form Definition (CRTFORMDF) command, CRTFORMDF syntax diagram
- Create Functional Area (CRTFCNARA) command, CRTFCNARA syntax diagram
- Create Graph Format (CRTGPHFMT) command, CRTGPHFMT syntax diagram
- Create Graph Package (CRTGPHPKG) command, CRTGPHPKG syntax diagram
- Create Graphics Symbol Set (CRTGSS) command, CRTGSS syntax diagram
- Create Historical Data (CRTHSTDTA) command, CRTHSTDTA syntax diagram
- Create Image Catalog (CRTIMGCLG) command, CRTIMGCLG syntax diagram
- Create Intersystem Communications Function File (CRTICFF) command, CRTICFF syntax diagram
- Create Java Program (CRTJVAPGM) command, CRTJVAPGM syntax diagram
- Create Job Description (CRTJOBDB) command, CRTJOBDB syntax diagram
- Create Job Queue (CRTJOBQ) command, CRTJOBQ syntax diagram
- Create Journal (CRTJRN) command, CRTJRN syntax diagram
- Create Journal Receiver (CRTJRNRCV) command, CRTJRNRCV syntax diagram
- Create Library (CRTLIB) command, CRTLIB syntax diagram
- Create Line Description (Async) (CRTLINASC) command, CRTLINASC syntax diagram
- Create Line Description (BSC) (CRTLINBSC) command, CRTLINBSC syntax diagram
- Create Line Description (DDI Network) (CRTLINDDI) command, CRTLINDDI syntax diagram
- Create Line Description (Ethernet) (CRTLINETH) command, CRTLINETH syntax diagram
- Create Line Description (Fax) (CRTLINFAX) command, CRTLINFAX syntax diagram
- Create Line Description (Frame Relay Network) (CRTLINFR) command, CRTLINFR syntax diagram
- Create Line Description (IDLC) (CRTLINIDLC) command, CRTLINIDLC syntax diagram
- Create Line Description (Network) (CRTLINNET) command, CRTLINNET syntax diagram
- Create Line Description (PPP) (CRTLINPPP) command, CRTLINPPP syntax diagram
- Create Line Description (SDLC) (CRLINSDLC) command, CRLINSDLC syntax diagram
- Create Line Description (TDLC) (CRTLINTDLC) command, CRTLINTDLC syntax diagram

- Create Line Description (Token-Ring Network) (CRTLINTRN) command, CRTLINTRN syntax diagram
- Create Line Description (Wireless) (CRTLINWLS) command, CRTLINWLS syntax diagram
- Create Line Description (X.25) (CRTLINX25) command, CRTLINX25 syntax diagram
- Create Locale (CRTLOCALE) command, CRTLOCALE syntax diagram
- Create Logical File (CRTLF) command, CRTLF syntax diagram
- Create Menu (CRTMNU) command, CRTMNU syntax diagram
- Create Message File (CRTMSGF) command, CRTMSGF syntax diagram
- Create Message File Menu (CRTMSGFMNU) command, CRTMSGFMNU syntax diagram
- Create Message Queue (CRTMSGQ) command, CRTMSGQ syntax diagram
- Create Mode Description (CRTMODD) command, CRTMODD syntax diagram
- Create NetBIOS Description (CRTNTBD) command, CRTNTBD syntax diagram
- Create NetWare Volume (CRTNTWVOL) command, CRTNTWVOL syntax diagram
- Create Network Interface (ATM Network) (CRTNWIATM) command, CRTNWIATM syntax diagram
- Create Network Interface (Frame Relay Network) (CRTNWIFR) command, CRTNWIFR syntax diagram
- Create Network Interface Description for ISDN (CRTNWIISDN) command, CRTNWIISDN syntax diagram
- Create Network Server Description (CRTNWSD) command, CRTNWSD syntax diagram
- Create Network Server Storage Space (CRTNWSSTG) command, CRTNWSSTG syntax diagram
- Create Node Group (CRTNODGRP) command, CRTNODGRP syntax diagram
- Create Node List (CRTNODL) command, CRTNODL syntax diagram
- Create Output Queue (CRTOUTQ) command, CRTOUTQ syntax diagram
- Create Overlay (CRTOVL) command, CRTOVL syntax diagram
- Create Page Definition (CRTPAGDFN) command, CRTPAGDFN syntax diagram
- Create Page Segment (CRTPAGSEG) command, CRTPAGSEG syntax diagram
- Create Panel Group (CRTPNLGRP) command, CRTPNLGRP syntax diagram
- Create Performance Data (CRTPFRDTA) command, CRTPFRDTA syntax diagram
- Create Performance Explorer Data (CRTPEXDTA) command, CRTPEXDTA syntax diagram
- Create Physical File (CRTPF) command, CRTPF syntax diagram
- Create Print Descriptor Group (CRTPDG) command, CRTPDG syntax diagram
- Create Print Services Facility Configuration (CRTPSFCFG) command, CRTPSFCFG syntax diagram
- Create Printer File (CRTPRTF) command, CRTPRTF syntax diagram
- Create Product Definition (CRTPRDDFN) command, CRTPRDDFN syntax diagram
- Create Product Load (CRTPRDLOD) command, CRTPRDLOD syntax diagram
- Create Program (CRTPGM) command, CRTPGM syntax diagram
- Create Program Temporary Fix (CRTPTF) command, CRTPTF syntax diagram
- Create Program Temporary Fix Package (CRTPTFPKG) command, CRTPTFPKG syntax diagram
- Create Query Management Form (CRTQMFORM) command, CRTQMFORM syntax diagram
- Create Query Management Query (CRTQMQRY) command, CRTQMQRY syntax diagram
- Create Question-and-Answer Database (CRTQSTDB) command, CRTQSTDB syntax diagram
- Create Question-and-Answer Load (CRTQSTLOD) command, CRTQSTLOD syntax diagram
- Create Save File (CRTSAVF) command, CRTSAVF syntax diagram
- Create Search Index (CRTSCHIDX) command, CRTSCHIDX syntax diagram
- Create Service Program (CRTSRVPGM) command, CRTSRVPGM syntax diagram
- Create Source Physical File (CRTSRCPF) command, CRTSRCPF syntax diagram
- Create Spelling Aid Dictionary (CRTSPADCT) command, CRTSPADCT syntax diagram

- Create Structured Query Language Package (CRTSQLPKG) command, CRTSQLPKG syntax diagram
- Create Subsystem Description (CRTSBSD) command, CRTSBSD syntax diagram
- Create System/36 Display File (CRTS36DSPF) command, CRTS36DSPF syntax diagram
- Create System/36 Menu (CRTS36MNU) command, CRTS36MNU syntax diagram
- Create System/36 Message File (CRTS36MSGF) command, CRTS36MSGF syntax diagram
- Create Table (CRTTBL) command, CRTTBL syntax diagram
- Create Tape Category (CRTTAPCGY) command, CRTTAPCGY syntax diagram
- Create Tape File (CRTTAPF) command, CRTTAPF syntax diagram
- Create User Profile (CRTUSRPRF) command, CRTUSRPRF syntax diagram
- Create User-Defined File System (CRTUDFS) command, CRTUDFS syntax diagram
- Create Validation List (CRTVLDL) command, CRTVLDL syntax diagram
- Create Work Station Customizing Object (CRTWSCST) command, CRTWSCST syntax diagram

Back to the top

D

- Data (DATA) command, DATA syntax diagram
- Deallocate (DLCOBJ) command, DLCOBJ syntax diagram
- Declare CL Variable (DCL) command, DCL syntax diagram
- Declare File (DCLF) command, DCLF syntax diagram
- Decompress Object (DCPOBJ) command, DCPOBJ syntax diagram
- Delay Job (DLYJOB) command, DLYJOB syntax diagram
- Delete Alert (DLTALR) command, DLTALR syntax diagram
- Delete Alert Table (DLTALRTBL) command, DLTALRTBL syntax diagram
- Delete APAR Data (DLTAPARDTA) command, DLTAPARDTA syntax diagram
- Delete Authority Holder (DLTAUTHLR) command, DLTAUTHLR syntax diagram
- Delete Authorization List (DLTAUTL) command, DLTAUTL syntax diagram
- Delete Binding Directory (DLTBNDDIR) command, DLTBNDDIR syntax diagram
- Delete Change Request Description (DLTCRQD) command, DLTCRQD syntax diagram
- Delete Class (DLTCLS) command, DLTCLS syntax diagram
- Delete Class-of-Service Description (DLTCOSD) command, DLTCOSD syntax diagram
- Delete Cluster (DLTCLU) command, DLTCLU syntax diagram
- Delete Cluster Resource Group (DLTCRG) command, DLTCRG syntax diagram
- Delete Cluster Resource Group from Cluster (DLTCRGCLU) command, DLTCRGCLU syntax diagram
- Delete Command (DLTCMD) command, DLTCMD syntax diagram
- Delete Communications Side Information (DLTCSI) command, DLTCSI syntax diagram
- Delete Communications Trace (DLTCMNTRC) command, DLTCMNTRC syntax diagram
- Delete Configuration List (DLTCFGL) command, DLTCFGL syntax diagram
- Delete Connection List (DLTCNNL) command, DLTCNNL syntax diagram
- Delete Controller Description (DLTCTLD) command, DLTCTLD syntax diagram
- Delete Data Area (DLTDTAARA) command, DLTDTAARA syntax diagram
- Delete Data Dictionary (DLTDTADCT) command, DLTDTADCT syntax diagram
- Delete Data Queue (DLTDTAQ) command, DLTDTAQ syntax diagram
- Delete DBCS Conversion Dictionary (DLTIGCDCT) command, DLTIGCDCT syntax diagram
- Delete DBCS Font Table (DLTIGCTBL) command, DLTIGCTBL syntax diagram
- Delete Device Description (DLTDEVD) command, DLTDEVD syntax diagram

- Delete Diskette Label (DLTDKTLBL) command, DLTDKTLBL syntax diagram
- Delete Distribution (DLTDST) command, DLTDST syntax diagram
- Delete Distribution List (DLTDSTL) command, DLTDSTL syntax diagram
- Delete Document Library Object (DLTDLO) command, DLTDLO syntax diagram
- Delete Document List (DLTDOCL) command, DLTDOCL syntax diagram
- Delete Edit Description (DLTEDTD) command, DLTEDTD syntax diagram
- Delete File (DLTF) command, DLTF syntax diagram
- Delete Filter (DLTFTR) command, DLTFTR syntax diagram
- Delete Firewall Log (DLTFRWLOG) command, DLTFRWLOG syntax diagram
- Delete Font Resources (DLTFNTRSC) command, DLTFNTRSC syntax diagram
- Delete Font Table (DLTFNTTBL) command, DLTFNTTBL syntax diagram
- Delete Form Definition (DLTFORMDF) command, DLTFORMDF syntax diagram
- Delete Functional Area (DLTFCNARA) command, DLTFCNARA syntax diagram
- Delete Graph Format (DLTGPHFMT) command, DLTGPHFMT syntax diagram
- Delete Graph Package (DLTGPHPKG) command, DLTGPHPKG syntax diagram
- Delete Graphics Symbol Set (DLTGSS) command, DLTGSS syntax diagram
- Delete Historical Data (DLTHSTDTA) command, DLTHSTDTA syntax diagram
- Delete Image Catalog (DLTIMGCLG) command, DLTIMGCLG syntax diagram
- Delete IPX Description (DLTIPXD) command, DLTIPXD syntax diagram
- Delete Java Program (DLTJVAPGM) command, DLTJVAPGM syntax diagram
- Delete Job Description (DLTJOBDD) command, DLTJOBDD syntax diagram
- Delete Job Description (DLTJOBDD) command, DLTJOBDD syntax diagram
- Delete Job Queue (DLTJOBQ) command, DLTJOBQ syntax diagram
- Delete Journal (DLTJRN) command, DLTJRN syntax diagram
- Delete Journal Receiver (DLTJRNRCV) command, DLTJRNRCV syntax diagram
- Delete Library (DLTLIB) command, DLTLIB syntax diagram
- Delete Licensed Program (DLTLICPGM) command, DLTLICPGM syntax diagram
- Delete Line Description (DLTLIND) command, DLTLIND syntax diagram
- Delete Locale (DLTLOCALE) command, DLTLOCALE syntax diagram
- Delete Locale (DLTMGTCOL) command, DLTMGTCOL syntax diagram
- Delete Media Definition (DLTMEDDFN) command, DLTMEDDFN syntax diagram
- Delete Menu (DLTMNU) command, DLTMNU syntax diagram
- Delete Message File (DLTMSGF) command, DLTMSGF syntax diagram
- Delete Message Queue (DLTMSGQ) command, DLTMSGQ syntax diagram
- Delete Mode Description (DLTMODD) command, DLTMODD syntax diagram
- Delete Module (DLTMOD) command, DLTMOD syntax diagram
- Delete NetBIOS Description (DLTNTBD) command, DLTNTBD syntax diagram
- Delete Network File (DLTNETF) command, DLTNETF syntax diagram
- Delete Network Interface Description (DLTNWID) command, DLTNWID syntax diagram
- Delete Network Server Description (DLTNWSD) command, DLTNWSD syntax diagram
- Delete Network Server Storage Space (DLTNWSSTG) command, DLTNWSSTG syntax diagram
- Delete Node Group (DLTNODGRP) command, DLTNODGRP syntax diagram
- Delete Node List (DLTNODL) command, DLTNODL syntax diagram
- Delete Output Queue (DLTOUTQ) command, DLTOUTQ syntax diagram
- Delete Overlay (DLTOVL) command, DLTOVL syntax diagram

- Delete Override (DLTOVR) command, DLTOVR syntax diagram
- Delete Override Device Entry (DLTOVRDEVE) command, DLTOVRDEVE syntax diagram
- Delete Page Definition (DLTPAGDFN) command, DLTPAGDFN syntax diagram
- Delete Page Segment (DLTPAGSEG) command, DLTPAGSEG syntax diagram
- Delete Panel Group (DLTPNLGRP) command, DLTPNLGRP syntax diagram
- Delete Performance Data (DLTPFRDTA) command, DLTPFRDTA syntax diagram
- Delete Performance Explorer Data (DLTPEXDTA) command, DLTPEXDTA syntax diagram
- Delete Print Descriptor Group (DLTPDG) command, DLTPDG syntax diagram
- Delete Print Services Facility Configuration (DLTPSFCFG) command, DLTPSFCFG syntax diagram
- Delete Problem (DLTPRB) command, DLTPRB syntax diagram
- Delete Product Definition (DLTPRDDFN) command, DLTPRDDFN syntax diagram
- Delete Product Load (DLTPRDLOD) command, DLTPRDLOD syntax diagram
- Delete Program (DLTPGM) command, DLTPGM syntax diagram
- Delete PTF (DLTPTF) command, DLTPTF syntax diagram
- Delete Query (DLTQRY) command, DLTQRY syntax diagram
- Delete Query Management Form (DLTQMFORM) command, DLTQMFORM syntax diagram
- Delete Query Management Query (DLTQMQRy) command, DLTQMQRy syntax diagram
- Delete Question (DLTQST) command, DLTQST syntax diagram
- Delete Question-and-Answer Database (DLTQSTDB) command, DLTQSTDB syntax diagram
- Delete Remote Program Temporary Fix (DLTRMTPTF) command, DLTRMTPTF syntax diagram
- Delete Search Index (DLTSCHIDX) command, DLTSCHIDX syntax diagram
- Delete Service Program (DLTSRVPGM) command, DLTSRVPGM syntax diagram
- Delete Spelling Aid Dictionary (DLTSPADCT) command, DLTSPADCT syntax diagram
- Delete Spooled File (DLTSPLF) command, DLTSPLF syntax diagram
- Delete Structured Query Language Package (DLTSQLPKG) command, DLTSQLPKG syntax diagram
- Delete Submitted Change Request (DLTSBMCRQ) command, DLTSBMCRQ syntax diagram
- Delete Subsystem Description (DLTSBSD) command, DLTSBSD syntax diagram
- Delete System Manager Object (DLTSMGOBJ) command, DLTSMGOBJ syntax diagram
- Delete Table (DLTTBL) command, DLTTBL syntax diagram
- Delete Tape Category (DLTTAPCGY) command, DLTTAPCGY syntax diagram
- Delete Trace Data (DLTTRC) command, DLTTRC syntax diagram
- Delete User Index (DLTUSRIDX) command, DLTUSRIDX syntax diagram
- Delete User Profile (DLTUSRPRF) command, DLTUSRPRF syntax diagram
- Delete User Queue (DLTUSRQ) command, DLTUSRQ syntax diagram
- Delete User Space (DLTUSRSPC) command, DLTUSRSPC syntax diagram
- Delete User Trace Buffer (DLTUSRTRC) command, DLTUSRTRC syntax diagram
- Delete User-Defined File System (DLTUDFS) command, DLTUDFS syntax diagram
- Delete Validation List (DLTVLDL) command, DLTVLDL syntax diagram
- Delete Work Station Customizing Object (DLTWSCST) command, DLTWSCST syntax diagram
- Disconnect Job (DSCJOB) command, DSCJOB syntax diagram
- Display Access Code (DSPACC) command, DSPACC syntax diagram
- Display Access Code Authority (DSPACCAUT) command, DSPACCAUT syntax diagram
- Display Access Group (DSPACCGRP) command, DSPACCGRP syntax diagram
- Display Activation Schedule (DSPACTSCD) command, DSPACTSCD syntax diagram
- Display Active Prestart Jobs (DSPACTPJ) command, DSPACTPJ syntax diagram

- Display Active Profile List (DSPACTPRFL) command, DSPACTPRFL syntax diagram
- Display APPN Information (DSPAPPNINF) command, DSPAPPNINF syntax diagram
- Display ASP Information (DSPASPBRM) command, DSPASPBRM syntax diagram
- Display Audit Journal Entries (DSPAUDJRNE) command, DSPAUDJRNE syntax diagram
- Display Authority (DSPAUT) command, DSPAUT syntax diagram
- Display Authority Holder (DSPAUTHLR) command, DSPAUTHLR syntax diagram
- Display Authorization List (DSPAUTL) command, DSPAUTL syntax diagram
- Display Authorization List Document Library Objects (DSPAUTLDLO) command, DSPAUTLDLO syntax diagram
- Display Authorization List Objects (DSPAUTLOBJ) command, DSPAUTLOBJ syntax diagram
- Display Authorized Users (DSPAUTUSR) command, DSPAUTUSR syntax diagram
- Display Backup List (DSPBCKUPL) command, DSPBCKUPL syntax diagram
- Display Backup Options (DSPBCKUP) command, DSPBCKUP syntax diagram
- Display Backup Plan Using BRM (DSPBKUBRM) command, DSPBKUBRM syntax diagram
- Display Backup Status (DSPBCKSTS) command, DSPBCKSTS syntax diagram
- Display Binding Directory (DSPBNDDIR) command, DSPBNDDIR syntax diagram
- Display Breakpoints (DSPBKP) command, DSPBKP syntax diagram
- Display Change Control Server Attributes (DSPCCSA) command, DSPCCSA syntax diagram
- Display Check Pending Constraint (DSPCPCST) command, DSPCPCST syntax diagram
- Display Class (DSPCLS) command, DSPCLS syntax diagram
- Display Class-of-Service Description (DSPCOSD) command, DSPCOSD syntax diagram
- Display Cluster Information (DSPCLUINF) command, DSPCLUINF syntax diagram
- Display Cluster Resource Group Information (DSPCRGINF) command, DSPCRGINF syntax diagram
- Display Coded Font (DSPCDEFNT) command, DSPCDEFNT syntax diagram
- Display Command (DSPCMD) command, DSPCMD syntax diagram
- Display Communications Side Information (DSPCSI) command, DSPCSI syntax diagram
- Display Configuration List (DSPCFGL) command, DSPCFGL syntax diagram
- Display Connection List (DSPCNL) command, DSPCNL syntax diagram
- Display Connection Status (DSPCNNSTS) command, DSPCNNSTS syntax diagram
- Display Controller Description (DSPCTLD) command, DSPCTLD syntax diagram
- Display Current Directory (DSPCURDIR) command, DSPCURDIR syntax diagram
- Display Data Area (DSPDTAARA) command, DSPDTAARA syntax diagram
- Display Data Dictionary (DSPDTADCT) command, DSPDTADCT syntax diagram
- Display Database Relations (DSPDBR) command, DSPDBR syntax diagram
- Display DBCS Conversion Dictionary (DSPIGCDCT) command, DSPIGCDCT syntax diagram
- Display Debug (DSPDBG) command, DSPDBG syntax diagram
- Display Debug Watches (DSPDBGWCH) command, DSPDBGWCH syntax diagram
- Display Device Description (DSPDEVD) command, DSPDEVD syntax diagram
- Display Directory Entries (DSPDIRE) command, DSPDIRE syntax diagram
- Display Diskette (DSPDKT) command, DSPDKT syntax diagram
- Display Distributed Data Management File (DSPDDMF) command, DSPDDMF syntax diagram
- Display Distribution Catalog Entry (DSPDSTCLGE) command, DSPDSTCLGE syntax diagram
- Display Distribution List (DSPDSTL) command, DSPDSTL syntax diagram
- Display Distribution Log (DSPDSTLOG) command, DSPDSTLOG syntax diagram
- Display Distribution Services (DSPDSTSRV) command, DSPDSTSRV syntax diagram

- Display Document (DSPDOC) command, DSPDOC syntax diagram
- Display Document Library Object Audit (DSPDLOAUD) command, DSPDLOAUD syntax diagram
- Display Document Library Object Authority (DSPDLOAUT) command, DSPDLOAUT syntax diagram
- Display Document Library Object Name (DSPDLONAM) command, DSPDLONAM syntax diagram
- Display Duplicate Media (DSPDUPBRM) command, DSPDUPBRM syntax diagram
- Display Edit Description (DSPEDTD) command, DSPEDTD syntax diagram
- Display Expiration Schedule (DSPEXPSCD) command, DSPEXPSCD syntax diagram
- Display Extended Wireless Controller Bar Code Entry (DSPEWCBCDE) command, DSPEWCBCDE syntax diagram
- Display Extended Wireless Controller Member (DSPEWCM) command, DSPEWCM syntax diagram
- Display Extended Wireless Controller PTC Entry (DSPEWCPTCE) command, DSPEWCPTCE syntax diagram
- Display Extended Wireless Line Member (DSPEWLM) command, DSPEWLM syntax diagram
- Display File (DSPF) command, DSPF syntax diagram
- Display File Description (DSPFD) command, DSPFD syntax diagram
- Display File Field Description (DSPFFD) command, DSPFFD syntax diagram
- Display Folder (DSPFLR) command, DSPFLR syntax diagram
- Display Font Resource Attributes (DSPFNTRSCA) command, DSPFNTRSCA syntax diagram
- Display Font Table (DSPFNTTBL) command, DSPFNTTBL syntax diagram
- Display Hardware Resources (DSPHDWRSC) command, DSPHDWRSC syntax diagram
- Display Help Document (DSPHLPDOC) command, DSPHLPDOC syntax diagram
- Display Hierarchical File Systems (DSPHFS) command, DSPHFS syntax diagram
- Display Historical Graph (DSPHSTGPH) command, DSPHSTGPH syntax diagram
- Display History Using Job Scheduler (DSPHSTJS) command, DSPHSTJS syntax diagram
- Display IPL Attributes (DSPIPLA) command, DSPIPLA syntax diagram
- Display IPX Description (DSPIPXD) command, DSPIPXD syntax diagram
- Display Java Program (DSPJVAPGM) command, DSPJVAPGM syntax diagram
- Display Job (DSPJOB) command, DSPJOB syntax diagram
- Display Job Description (DSPJOBDB) command, DSPJOBDB syntax diagram
- Display Job Log (DSPJOBLOG) command, DSPJOBLOG syntax diagram
- Display Job Tables (DSPJOBTL) command, DSPJOBTL syntax diagram
- Display Job Using Job Scheduler (DSPJOBJS) command, DSPJOBJS syntax diagram
- Display Journal (DSPJRN) command, DSPJRN syntax diagram
- Display Journal Receiver Attributes (DSPJRNRCVA) command, DSPJRNRCVA syntax diagram
- Display Keyboard Map (DSPKBDMAP) command, DSPKBDMAP syntax diagram
- Display LAN Media Library (DSPLANMLB) command, DSPPLANMLB syntax diagram
- Display Library (DSPLIB) command, DSPLIB syntax diagram
- Display Library Description (DSPLIBD) command, DSPLIBD syntax diagram
- Display Library List (DSPLIBL) command, DSPLIBL syntax diagram
- Display License Key Information (DSPLICKEY) command, DSPLICKEY syntax diagram
- Display Line Description (DSPLIND) command, DSPLIND syntax diagram
- Display Local Area Network Adapter Profile (DSPLANADPP) command, DSPPLANADPP syntax diagram
- Display Local Area Network Status (DSPLANSTS) command, DSPPLANSTS syntax diagram
- Display Log (DSPLOG) command, DSPLOG syntax diagram
- Display Log for BRM (DSPLOGBRM) command, DSPLOGBRM syntax diagram

- Display Log for Job Scheduler (DSPLOGJS) command, DSPLOGJS syntax diagram
- Display Managed System Attributes (DSPMGDSYSA) command, DSPMGDSYSA syntax diagram
- Display Menu Attributes (DSPMNUA) command, DSPMNUA syntax diagram
- Display Message Descriptions (DSPMSGD) command, DSPMSGD syntax diagram
- Display Messages (DSPMSG) command, DSPMSG syntax diagram
- Display Mode Description (DSPMODD) command, DSPMODD syntax diagram
- Display Mode Status (DSPMODSTS) command, DSPMODSTS syntax diagram
- Display Module (DSPMOD) command, DSPMOD syntax diagram
- Display Module Source (DSPMODSRC) command, DSPMODSRC syntax diagram
- Display Mounted File System Information (DSPMFSINF) command, DSPMFSINF syntax diagram
- Display Mounted File System Information (STATFS) command, STATFS syntax diagram
- Display NetBIOS Description (DSPNTBD) command, DSPNTBD syntax diagram
- Display Network Attributes (DSPNETA) command, DSPNETA syntax diagram
- Display Network Interface Description (DSPNWID) command, DSPNWID syntax diagram
- Display Network Server Attributes (DSPNWSA) command, DSPNWSA syntax diagram
- Display Network Server Description (DSPNWS) command, DSPNWS syntax diagram
- Display Network Server Storage Space (DSPNWSSTG) command, DSPNWSSTG syntax diagram
- Display Network Server User Attributes (DSPNWSUSRA) command, DSPNWSUSRA syntax diagram
- Display Network Server Users (DSPNWSUSR) command, DSPNWSUSR syntax diagram
- Display Nickname (DSPNCK) command, DSPNCK syntax diagram
- Display Node Group (DSPNODGRP) command, DSPNODGRP syntax diagram
- Display Object Authority (DSPOBJAUT) command, DSPOBJAUT syntax diagram
- Display Object Description (DSPOBJD) command, DSPOBJD syntax diagram
- Display Object Links (DSPLNK) command, DSPLNK syntax diagram
- Display Optical (DSPOPT) command, DSPOPT syntax diagram
- Display Optical Locks (DSPOPTLCK) command, DSPOPTLCK syntax diagram
- Display Optical Server (DSPOPTSVR) command, DSPOPTSVR syntax diagram
- Display OptiConnect Link Status (DSPOPCLNK) command, DSPOPCLNK syntax diagram
- Display Override (DSPOVR) command, DSPOVR syntax diagram
- Display Performance Data (DSPPFRDTA) command, DSPPFRDTA syntax diagram
- Display Performance Graph (DSPPFRGPH) command, DSPPFRGPH syntax diagram
- Display Physical File Member (DSPPFM) command, DSPPFM syntax diagram
- Display Power On/Off Schedule (DSPPWRSCD) command, DSPPWRSCD syntax diagram
- Display Print Descriptor Group Profile (DSPPDGPRF) command, DSPPDGPRF syntax diagram
- Display Print Services Facility Configuration (DSPPSFCFG) command, DSPPSFCFG syntax diagram
- Display Problem (DSPPRB) command, DSPPRB syntax diagram
- Display Program (DSPPGM) command, DSPPGM syntax diagram
- Display Program References (DSPPGMREF) command, DSPPGMREF syntax diagram
- Display Program Temporary Fix (DSPPTF) command, DSPPTF syntax diagram
- Display Program Variable (DSPPGMVAR) command, DSPPGMVAR syntax diagram
- Display Programs that Adopt (DSPPGMADP) command, DSPPGMADP syntax diagram
- Display PTF Cover Letter (DSPPTFCVR) command, DSPPTFCVR syntax diagram
- Display Received Commands (DSPRCVCM) command, DSPRCVCM syntax diagram
- Display Record Locks (DSPRCDLCK) command, DSPRCDLCK syntax diagram
- Display Recovery for Access Paths (DSPRCYAP) command, DSPRCYAP syntax diagram

- Display Relational Database Directory Entry (DSPRDBDIRE) command, DSPRDBDIRE syntax diagram
- Display Remote Definition (DSPRMTDFN) command, DSPRMTDFN syntax diagram
- Display Save File (DSPSAVF) command, DSPSAVF syntax diagram
- Display Security Attributes (DSPSECA) command, DSPSECA syntax diagram
- Display Security Auditing Values (DSPSECAUD) command, DSPSECAUD syntax diagram
- Display Server Authentication Entries (DSPSVRAUTE) command, DSPSVRAUTE syntax diagram
- Display Service Attributes (DSPSRVA) command, DSPSRVA syntax diagram
- Display Service Program (DSPSRVPG) command, DSPSRVPG syntax diagram
- Display Service Provider Attributes (DSPSRVPVDA) command, DSPSRVPVDA syntax diagram
- Display Service Status (DSPSRVSTS) command, DSPSRVSTS syntax diagram
- Display Software Resources (DSPSFWRSC) command, DSPSFWRSC syntax diagram
- Display Sphere of Control Status (DPSOCSTS) command, DPSOCSTS syntax diagram
- Display Spooled File (DPSPLF) command, DPSPLF syntax diagram
- Display Submitted Change Request Activities (DPSBMCRQA) command, DPSBMCRQA syntax diagram
- Display Submitted Change Request Messages (DPSBMCRQM) command, DPSBMCRQM syntax diagram
- Display Submitted Change Requests (DPSBMCRQ) command, DPSBMCRQ syntax diagram
- Display Subsystem Description (DPSBSD) command, DPSBSD syntax diagram
- Display System Status (DPSYSSTS) command, DPSYSSTS syntax diagram
- Display System Value (DPSYSVAL) command, DPSYSVAL syntax diagram
- Display System/36 (DPS36) command, DPS36 syntax diagram
- Display Tape (DSPTAP) command, DSPTAP syntax diagram
- Display Tape Cartridge (DSPTAPCTG) command, DSPTAPCTG syntax diagram
- Display Tape Category (DSPTAPCGY) command, DSPTAPCGY syntax diagram
- Display Tape Status (DSPTAPSTS) command, DSPTAPSTS syntax diagram
- Display Trace (DSPTRC) command, DSPTRC syntax diagram
- Display Trace Data (DSPTRCDTA) command, DSPTRCDTA syntax diagram
- Display Trademarks (DSPTM) command, DSPTM syntax diagram
- Display User Permission (DSPUSRPMN) command, DSPUSRPMN syntax diagram
- Display User Print Information (DSPUSRPTI) command, DSPUSRPTI syntax diagram
- Display User Profile (DSPUSRPRF) command, DSPUSRPRF syntax diagram
- Display User-Defined File System (DSPUDFS) command, DSPUDFS syntax diagram
- Display Work Station User (DSPWSUSR) command, DSPWSUSR syntax diagram
- Do (DO) command, DO syntax diagram
- Dump (DMP) command, DMP syntax diagram
- Dump BRM (DMPBRM) command, DMPBRM syntax diagram
- Dump Cluster Trace (DMPCLUTRC) command, DMPCLUTRC syntax diagram
- Dump Control Language Program (DMPCLPGM) command, DMPCLPGM syntax diagram
- Dump Document Library Object (DMPDLO) command, DMPDLO syntax diagram
- Dump Java Virtual Machine (DMPJVM) command, DMPJVM syntax diagram
- Dump Job (DMPJOB) command, DMPJOB syntax diagram
- Dump Job Internal (DMPJOBINT) command, DMPJOBINT syntax diagram
- Dump Object (DMPOBJ) command, DMPOBJ syntax diagram
- Dump System Object (DMPSYSOBJ) command, DMPSYSOBJ syntax diagram

- Dump Tape (DMPTAP) command, DMPTAP syntax diagram
- Dump Trace (DMPTRC) command, DMPTRC syntax diagram
- Dump User Trace Buffer (DMPUSRTRC) command, DMPUSRTRC syntax diagram
- Duplicate Diskette (DUPDKT) command, DUPDKT syntax diagram
- Duplicate Media Using BRM (DUPMEDBRM) command, DUPMEDBRM syntax diagram
- Duplicate Optical (DUPOPT) command, DUPOPT syntax diagram
- Duplicate Tape (DUPTAP) command, DUPTAP syntax diagram

Back to the top

E

- Edit Authorization List (EDTAUTL) command, EDTAUTL syntax diagram
- Edit Backup Lis (EDTBCKUPL) command, EDTBCKUPL syntax diagram
- Edit Check Pending Constraints (EDTCPCST) command, EDTCPCST syntax diagram
- Edit DBCS Conversion Dictionary (EDTIGCDCT) command, EDTIGCDCT syntax diagram
- Edit Document (EDTDOC) command, EDTDOC syntax diagram
- Edit Document Library Object Authority (EDTDLOAU) command, EDTDLOAU syntax diagram
- Edit File (EDTF) command, EDTF syntax diagram
- Edit Library List (EDTLIBL) command, EDTLIBL syntax diagram
- Edit Object Authority (EDTOBJAUT) command, EDTOBJAUT syntax diagram
- Edit Questions and Answers (EDTQST) command, EDTQST syntax diagram
- Edit Rebuild of Access Paths (EDTRBDAP) command, EDTRBDAP syntax diagram
- Edit Recovery for Access Paths (EDTRCYAP) command, EDTRCYAP syntax diagram
- Edit System/36 Procedure Attributes (EDTS36PRCA) command, EDTS36PRCA syntax diagram
- Edit System/36 Program Attributes (EDTS36PGMA) command, EDTS36PGMA syntax diagram
- Edit System/36 Source Attributes (EDTS36SRCA) command, EDTS36SRCA syntax diagram
- Edit Workstation Object Authority (EDTWSOAUT) command, EDTWSOAUT syntax diagram
- Eject Emulation Output (EJTEMLOUT) command, EJTEMLOUT syntax diagram
- Else (ELSE) command, ELSE syntax diagram
- Emulate Printer Key (EMLPRTKEY) command, EMLPRTKEY syntax diagram
- End ASP Balance (ENDASPBAL) command, ENDASPBAL syntax diagram
- End Batch Job (ENDBCHJOB) command, ENDBCHJOB syntax diagram
- End Cleanup (ENDCLNUP) command, ENDCLNUP syntax diagram
- End Cluster Node (ENDCLUNOD) command, ENDCLUNOD syntax diagram
- End Cluster Resource Group (ENDCRG) command, ENDCRG syntax diagram
- End Clustered Hash Table Server (ENDCHTSVR) command, ENDCHTSVR syntax diagram
- End Commitment Control (ENDCMTCTL) command, ENDCMTCTL syntax diagram
- End Communications Server (ENDCMNSVR) command, ENDCMNSVR syntax diagram
- End Communications Trace (ENDCMNTRC) command, ENDCMNTRC syntax diagram
- End Controller Recovery (ENDCTLRCY) command, ENDCTLRCY syntax diagram
- End Copy Screen (ENDCPYSC) command, ENDCPYSC syntax diagram
- End Database Monitor (ENDDBMON) command, ENDDBMON syntax diagram
- End Debug (ENDDDBG) command, ENDDDBG syntax diagram
- End Debug Server (ENDDBGSVR) command, ENDDBGSVR syntax diagram
- End Device Recovery (ENDDEVRCY) command, ENDDEVRCY syntax diagram
- End Directory Shadowing (ENDDIRSHD) command, ENDDIRSHD syntax diagram

- End Disk Reorganization (ENDDSKRGZ) command, ENDDSKRGZ syntax diagram
- End Do (ENDDO) command, ENDDO syntax diagram
- End Group Job (ENDGRPJOB) command, ENDGRPJOB syntax diagram
- End Host Server (ENDHOSTSVR) command, ENDHOSTSVR syntax diagram
- End HTTP Crawl (ENDHTTPCRL) command
- End Input (ENDINP) command, ENDINP syntax diagram
- End IP over SNA Interface (ENDIPSIFC) command, ENDIPSIFC syntax diagram
- End Job (ENDJOB) command, ENDJOB syntax diagram
- End Job Abnormal (ENDJOBABN) command, ENDJOBABN syntax diagram
- End Job Scheduler (ENDJS) command, ENDJS syntax diagram
- End Job Trace (ENDJOBTRC) command, ENDJOBTRC syntax diagram
- End Journal (ENDJRN) command, ENDJRN syntax diagram
- End Journal Access Path (ENDJRNAP) command, ENDJRNAP syntax diagram
- End Journal Object (ENDJRNOBJ) command, ENDJRNOBJ syntax diagram
- End Journal Physical File (ENDJRNPF) command, ENDJRNPF syntax diagram
- End Line Recovery (ENDLINRCY) command, ENDLINRCY syntax diagram
- End Mail Server Framework (ENDMSF) command, ENDMSF syntax diagram
- End Managed System Services (ENDMGDSYS) command, ENDMGDSYS syntax diagram
- End Manager Services (ENDMGRSRV) command, ENDMGRSRV syntax diagram
- End Mode (ENDMOD) command, ENDMOD syntax diagram
- End Network File System Server (ENDNFSSVR) command, ENDNFSSVR syntax diagram
- End Network Interface Recovery (ENDNWIRCY) command, ENDNWIRCY syntax diagram
- End Pass-Through (ENDPASTHR) command, ENDPASTHR syntax diagram
- End Performance Explorer (ENDPEX) command, ENDPEX syntax diagram
- End Performance Trace (ENDPFRTRC) command, ENDPFRTRC syntax diagram
- End Point-to-Point TCP/IP (ENDTCPPTP) command, ENDTCPPTP syntax diagram
- End Prestart Jobs (ENDPJ) command, ENDPJ syntax diagram
- End Printer Emulation (ENDPRTEML) command, ENDPRTEML syntax diagram
- End Program (ENDPGM) command, ENDPGM syntax diagram
- End Program Profiling (ENDPGMPRF) command, ENDPGMPRF syntax diagram
- End Reader (ENDRDR) command, ENDRDR syntax diagram
- End Receive (ENDRCV) command, ENDRCV syntax diagram
- End Remote Support (ENDRMTSPT) command, ENDRMTSPT syntax diagram
- End Request (ENDRQS) command, ENDRQS syntax diagram
- End RPC Binder Daemon (ENDRPCBIND) command, ENDRPCBIND syntax diagram
- End Service Job (ENDSRVJOB) command, ENDSRVJOB syntax diagram
- End Submitted Change Request Activity (ENDSBMCRQA) command, ENDSBMCRQA syntax diagram
- End Subsystem (ENDSBS) command, ENDSBS syntax diagram
- End System (ENDSYS) command, ENDSYS syntax diagram
- End System Manager (ENDSYSMGR) command, ENDSYSMGR syntax diagram
- End System/36 (ENDS36) command, ENDS36 syntax diagram
- End TCP/IP (ENDTCP) command, ENDTCP syntax diagram
- End TCP/IP Abnormal (ENDTCPABN) command, ENDTCPABN syntax diagram
- End TCP/IP Connection (ENDTCPCNN) command, ENDTCPCNN syntax diagram
- End TCP/IP Interface (ENDTCPIFC) command, ENDTCPIFC syntax diagram

- End TCP/IP Server (ENDTCPSVR) command, ENDTCPSVR syntax diagram
- End Technical Information Exchange Session (ENDTIESSN) command, ENDTIESSN syntax diagram
- End Trace (ENDTRC) command, ENDTRC syntax diagram
- End Trap Manager (ENDTRPMGR) command, ENDTRPMGR syntax diagram
- End Ultimedia System Facilities (ENDUSF) command, ENDUSF syntax diagram
- End Writer (ENDWTR) command, ENDWTR syntax diagram
- Extract Media Information (EXTMEDIBRM) command, EXTMEDIBRM syntax diagram

[Back to the top](#)

F

- File Document (FILDOC) command, FILDOC syntax diagram

[Back to the top](#)

G

- Generate License Key (GENLICKKEY) command, GENLICKKEY syntax diagram
- Go To (GOTO) command, GOTO syntax diagram
- Go to Menu (GO) command, GO syntax diagram
- Grant Access Code Authority (GRTACCAUT) command, GRTACCAUT syntax diagram
- Grant Object Authority (GRTOBJAUT) command, GRTOBJAUT syntax diagram
- Grant User Authority (GRTUSRAUT) command, GRTUSRAUT syntax diagram
- Grant User Permission (GRTUSRPMN) command, GRTUSRPMN syntax diagram
- Grant Workstation Object Authority (GRTWSOAUT) command, GRTWSOAUT syntax diagram

[Back to the top](#)

H

- Hold Communications Device (HLDCMNDEV) command, HLDCMNDEV syntax diagram
- Hold Distribution Queue (HLDDSTQ) command, HLDDSTQ syntax diagram
- Hold Job (HLDJOB) command, HLDJOB syntax diagram
- Hold Job Queue (HLDJOBQ) command, HLDJOBQ syntax diagram
- Hold Job Schedule Entry (HLDJOBSCDE) command, HLDJOBSCDE syntax diagram
- Hold Job Using Job Scheduler (HLDJOBJS) command, HLDJOBJS syntax diagram
- Hold Output Queue (HLDOUTQ) command, HLDOUTQ syntax diagram
- Hold Program Temporary Fix (HLDPTF) command, HLDPTF syntax diagram
- Hold Reader (HLDRDR) command, HLDRDR syntax diagram
- Hold Spooled File (HLDSPLF) command, HLDSPLF syntax diagram
- Hold Submitted Change Request Activity (HLDSBMCRQA) command, HLDSBMCRQA syntax diagram
- Hold Writer (HLDWTR) command, HLDWTR syntax diagram

[Back to the top](#)

I

- If (IF) command, IF syntax diagram
- Initialize BRMS/400 (INZBRM) command, INZBRM syntax diagram
- Initialize Client Access (INZPCS) command, INZPCS syntax diagram
- Initialize Diskette (INZDKT) command, INZDKT syntax diagram

- Initialize Distribution Queue (INZDSTQ) command, INZDSTQ syntax diagram
- Initialize Media Using BRM (INZMEDBRM) command, INZMEDBRM syntax diagram
- Initialize Optical (INZOPT) command, INZOPT syntax diagram
- Initialize Physical File Member (INZPFM) command, INZPFM syntax diagram
- Initialize System (INZSYS) command, INZSYS syntax diagram
- Initialize Tape (INZTAP) command, INZTAP syntax diagram
- Install Program Temporary Fix (INSPTF) command, INSPTF syntax diagram
- Install Remote Product (INSRMTPRD) command, INSRMTPRD syntax diagram

Back to the top

L

- Link Data Definition (LNKDTADFN) command, LNKDTADFN syntax diagram
- Load and Run Media Program (LODRUN) command, LODRUN syntax diagram
- Load or Unload Image Catalog (LODIMGCLG) command, LODIMGCLG syntax diagram
- Load Program Temporary Fix (LODPTF) command, LODPTF syntax diagram
- Load Question-and-Answer Database (LODQSTDB) command, LODQSTDB syntax diagram

Back to the top

M

- Merge Message Catalog (GENCAT) command, GENCAT syntax diagram
- Merge Message Catalog (MRGMSGCLG) command, MRGMSGCLG syntax diagram
- Merge Message File (MRGMSGF) command, MRGMSGF syntax diagram
- Merge TCP/IP Host Table (MRGTCPHT) command, MRGTCPHT syntax diagram
- Migrate Using BRM (MGRBRM) command, MGRBRM syntax diagram
- Monitor Message (MONMSG) command, MONMSG syntax diagram
- Monitor Save While Active (MONSWABRM) command, MONSWABRM syntax diagram
- Move (MOV) command, MOV syntax diagram
- Move (MOVE) command, MOVE syntax diagram
- Move Document (MOVDOC) command, MOVDOC syntax diagram
- Move Media Using BRM (MOVMEDBRM) command, MOVMEDBRM syntax diagram
- Move Object (MOV OBJ) command, MOV OBJ syntax diagram
- Move Saved Spooled Files Using BRM (MOV SPLFBRM) command, MOV SPLFBRM syntax diagram

Back to the top

N

- NETSTAT (NETSTAT) command, NETSTAT syntax diagram

Back to the top

O

- Open Query File (OPNQRYF) command, OPNQRYF syntax diagram
- Order Supported Product PTFs (ORDSPTPTF) command, ORDSPTPTF syntax diagram
- Override with Database File (OVRDBF) command, OVRDBF syntax diagram
- Override with Diskette File (OVRDKTF) command, OVRDKTF syntax diagram
- Override with Display File (OVRDSPF) command, OVRDSPF syntax diagram

- Override with Intersystem Communications Function File (OVRICFF) command, OVRICFF syntax diagram
- Override with Intersystem Communications Function Program Device Entry (OVRICFDEVE) command, OVRICFDEVE syntax diagram
- Override with Message File (OVRMSGF) command, OVRMSGF syntax diagram
- Override with Printer File (OVRPRTF) command, OVRPRTF syntax diagram
- Override with Save File (OVRSAVF) command, OVRSAVF syntax diagram
- Override with Tape File (OVRTAPF) command, OVRTAPF syntax diagram

Back to the top

P

- Package Installable Object (PKGINSOBJ) command, PKGINSOBJ syntax diagram
- Package Product for Distribution (PKGPRDDST) command, PKGPRDDST syntax diagram
- Package Product Option (PKGPRDOPT) command, PKGPRDOPT syntax diagram
- PING (PING) command, PING syntax diagram
- Position Database File (POSDBF) command, POSDBF syntax diagram
- Power Down System (PWRDWNSYS) command, PWRDWNSYS syntax diagram
- Print Activity Report (PRTACTRPT) command, PRTACTRPT syntax diagram
- Print Adopting Objects (PRTADPOBJ) command, PRTADPOBJ syntax diagram
- Print Advanced Function Printer Data (PRTAFPDT) command, PRTAFPDT syntax diagram
- Print Command Usage (PRTCMDUSG) command, PRTCMDUSG syntax diagram
- Print Communications Security (PRTCMNSEC) command, PRTCMNSEC syntax diagram
- Print Communications Trace (PRTCMNTRC) command, PRTCMNTRC syntax diagram
- Print Component Report (PRTCPTTRPT) command, PRTCPTTRPT syntax diagram
- Print Device Addresses (PRTDEVADR) command, PRTDEVADR syntax diagram
- Print Disk Information (PRTDSKINF) command, PRTDSKINF syntax diagram
- Print Document (PRTDOC) command, PRTDOC syntax diagram
- Print Error Log (PRTERRLOG) command, PRTERRLOG syntax diagram
- Print Internal Data (PRTINTDTA) command, PRTINTDTA syntax diagram
- Print IP over SNA Configuration (PRTIPSCFG) command, PRTIPSCFG syntax diagram
- Print Job Description Authority (PRTJOBDAUT) command, PRTJOBDAUT syntax diagram
- Print Job Report (PRTJOBTRPT) command, PRTJOBTRPT syntax diagram
- Print Job Trace (PRTJOBTRC) command, PRTJOBTRC syntax diagram
- Print Labels Using BRM (PRTLBLBRM) command, PRTLBLBRM syntax diagram
- Print Lock Report (PRTLCKRPT) command, PRTLCKRPT syntax diagram
- Print Media Exceptions Using BRM (PRTMEDBRM) command, PRTMEDBRM syntax diagram
- Print Media Movement (PRTMOVBRM) command, PRTMOVBRM syntax diagram
- Print Performance Explorer Report (PRTPEXRPT) command, PRTPEXRPT syntax diagram
- Print Point-to-Point TCP/IP Profile (PRTTCPPTP) command, PRTTCPPTP syntax diagram
- Print Pool Report (PRTPOLRPT) command, PRTPOLRPT syntax diagram
- Print Private Authorities (PRTPVTAUT) command, PRTPVTAUT syntax diagram
- Print Profile Internals (PRTPRFINT) command, PRTPRFINT syntax diagram
- Print Publicly Authorized Objects (PRTPUBAUT) command, PRTPUBAUT syntax diagram
- Print Queue Authority (PRTQAUT) command, PRTQAUT syntax diagram
- Print Resource Report (PRTRSCRPT) command, PRTRSCRPT syntax diagram

- Print Schedule Using Job Scheduler (PRTSCDJS) command, PRTSCDJS syntax diagram
- Print Stop Word List (PRTSWL) command, PRTSWL syntax diagram
- Print Structured Query Language Information (PRTSQLINF) command, PRTSQLINF syntax diagram
- Print Subsystem Description Authority (PRTSBSDAUT) command, PRTSBSDAUT syntax diagram
- Print System Information (PRTSYSINF) command, PRTSYSINF syntax diagram
- Print System Report (PRTSYSRPT) command, PRTSYSRPT syntax diagram
- Print System Security Attributes (PRTSYSSECA) command, PRTSYSSECA syntax diagram
- Print Trace (PRTRC) command, PRTRC syntax diagram
- Print Trace Report (PRTRCRPT) command, PRTRCRPT syntax diagram
- Print Transaction Report (PRTTNSRPT) command, PRTTNSRPT syntax diagram
- Print Trigger Program (PRTTRGPGM) command, PRTTRGPGM syntax diagram
- Print User Objects (PRTUSROBJ) command, PRTUSROBJ syntax diagram
- Print User Profile (PRTUSRPRF) command, PRTUSRPRF syntax diagram
- Program (PGM) command, PGM syntax diagram

Back to the top

Q

- QSH (QSH) command, QSH syntax diagram
- Query Distribution (QRYDST) command, QRYDST syntax diagram
- Query Document Library (QRYDOCLIB) command, QRYDOCLIB syntax diagram
- Query Problem Status (QRYPRBSTS) command, QRYPRBSTS syntax diagram
- Query Technical Information Exchange File (QRYTIEF) command, QRYTIEF syntax diagram

Back to the top

R

- Receive Distribution (RCVDST) command, RCVDST syntax diagram
- Receive File (RCVF) command, RCVF syntax diagram
- Receive Journal Entry (RCVJRNE) command, RCVJRNE syntax diagram
- Receive Message (RCVMSG) command, RCVMSG syntax diagram
- Receive Network File (RCVNETF) command, RCVNETF syntax diagram
- Receive Technical Information Exchange File (RCVTIEF) command, RCVTIEF syntax diagram
- Reclaim Activation Group (RCLACTGRP) command, RCLACTGRP syntax diagram
- Reclaim Distributed Data Management Conversations (RCLDDMCNV) command, RCLDDMCNV syntax diagram
- Reclaim Document Library Object (RCLDLO) command, RCLDLO syntax diagram
- Reclaim Library (RCLLIB) command, RCLLIB syntax diagram
- Reclaim Optical (RCLOPT) command, RCLOPT syntax diagram
- Reclaim Resources (RCLRSC) command, RCLRSC syntax diagram
- Reclaim Spool Storage (RCLSPLSTG) command, RCLSPLSTG syntax diagram
- Reclaim Storage (RCLSTG) command, RCLSTG syntax diagram
- Reclaim Temporary Storage (RCLTMPSTG) command, RCLTMPSTG syntax diagram
- Release Communications Device (RLSCMNDEV) command, RLSCMNDEV syntax diagram
- Release Distribution Queue (RLSDSTQ) command, RLSDSTQ syntax diagram
- Release Integrated File System Locks (RLSIFSLCK) command, RLSIFSLCK syntax diagram
- Release Job (RLSJOB) command, RLSJOB syntax diagram

- Release Job Queue (RLSJQBQ) command, RLSJOBQ syntax diagram
- Release Job Schedule Entry (RLSJQBSCDE) command, RLSJOBSCDE syntax diagram
- Release Job Using Job Scheduler (RLSJQBJS) command, RLSJOBJS syntax diagram
- Release Output Queue (RLSOUTQ) command, RLSOUTQ syntax diagram
- Release Program Temporary Fix (RLSPTF) command, RLSPTF syntax diagram
- Release Reader (RLSRDR) command, RLSRDR syntax diagram
- Release Remote Phase (RLSRMTPHS) command, RLSRMTPHS syntax diagram
- Release Spooled File (RLSSPLF) command, RLSSPLF syntax diagram
- Release Submitted Change Request Activity (RLSSBMCRQA) command, RLSSBMCRQA syntax diagram
- Release Writer (RLSWTR) command, RLSWTR syntax diagram
- Remove Access Code (RMVACC) command, RMVACC syntax diagram
- Remove Alert Description (RMVALRD) command, RMVALRD syntax diagram
- Remove Authorization List Entry (RMVAUTLE) command, RMVAUTLE syntax diagram
- Remove Autostart Job Entry (RMVAJE) command, RMVAJE syntax diagram
- Remove Binding Directory Entry (RMVBNDIRE) command, RMVBNDIRE syntax diagram
- Remove Breakpoint (RMVBKP) command, RMVBKP syntax diagram
- Remove Change Control Server Client (RMVCCSCLT) command, RMVCCSCLT syntax diagram
- Remove Change Request Description Activity (RMVCRQDA) command, RMVCRQDA syntax diagram
- Remove Cluster Node Entry (RMVCLUNODE) command, RMVCLUNODE syntax diagram
- Remove Cluster Resource Group Device Entry (RMVCRGDEVE) command, RMVCRGDEVE syntax diagram
- Remove Cluster Resource Group Node Entry (RMVCRGNODE) command, RMVCRGNODE syntax diagram
- Remove Communications Entry (RMVCMNE) command, RMVCMNE syntax diagram
- Remove Community for SNMP (RMVCOMSNMP) command, RMVCOMSNMP syntax diagram
- Remove Configuration List Entries (RMVCFGLE) command, RMVCFGLE syntax diagram
- Remove Connection List Entry (RMVCNNLE) command, RMVCNNLE syntax diagram
- Remove Device Domain Entry (RMVDEVDMNE) command, RMVDEVDMNE syntax diagram
- Remove Director (RD) command, RD syntax diagram
- Remove Directory (RMDIR) command, RMDIR syntax diagram
- Remove Directory (RMVDIR) command, RMVDIR syntax diagram
- Remove Directory Entry (RMVDIRE) command, RMVDIRE syntax diagram
- Remove Directory Shadow System (RMVDIRSHD) command, RMVDIRSHD syntax diagram
- Remove Distribution Catalog Entry (RMVDSTCLGE) command, RMVDSTCLGE syntax diagram
- Remove Distribution List Entry (RMVDSTLE) command, RMVDSTLE syntax diagram
- Remove Distribution Queue (RMVDSTQ) command, RMVDSTQ syntax diagram
- Remove Distribution Route (RMVDSTRTE) command, RMVDSTRTE syntax diagram
- Remove Distribution Secondary System Name (RMVDSTSYSN) command, RMVDSTSYSN syntax diagram
- Remove Document Library Object Authority (RMVDLOAUT) command, RMVDLOAUT syntax diagram
- Remove Emulation Configuration Entry (RMVEMLCFGE) command, RMVEMLCFGE syntax diagram
- Remove Environment Variable (RMVENVVAR) command, RMVENVVAR syntax diagram
- Remove Exit Program (RMVEXITPGM) command, RMVEXITPGM syntax diagram
- Remove Extended Wireless Controller Bar Code Entry (RMVEWCBCDE) command, RMVEWCBCDE syntax diagram

- Remove Extended Wireless Controller PTC Entry (RMVEWCPTCE) command, RMVEWCPTCE syntax diagram
- Remove Filter Action Entry (RMVFTRACNE) command, RMVFTRACNE syntax diagram
- Remove Filter Selection Entry (RMVFTRSLTE) command, RMVFTRSLTE syntax diagram
- Remove Font Table Entry (RMVFNTTBLE) command, RMVFNTTBLE syntax diagram
- Remove History Using Job Scheduler (RMVHSTJS) command, RMVHSTJS syntax diagram
- Remove Image Catalog Entry (RMVIMGCLGE) command, RMVIMGCLGE syntax diagram
- Remove Intersystem Communications Function Program Device Entry (RMVICFDEVE) command, RMVICFDEVE syntax diagram
- Remove IP over SNA interface (RMVIPSIFC) command, RMVIPSIFC syntax diagram
- Remove IP over SNA Location Entry (RMVIPSLOC) command, RMVIPSLOC syntax diagram
- Remove IP over SNA Route (RMVIPS RTE) command, RMVIPS RTE syntax diagram
- Remove Job Queue Entry (RMVJOBQE) command, RMVJOBQE syntax diagram
- Remove Job Schedule Entry (RMVJOBSCDE) command, RMVJOBSCDE syntax diagram
- Remove Job Using Job Scheduler (RMVJOBJS) command, RMVJOBJS syntax diagram
- Remove Journalized Changes (RMVJRNCHG) command, RMVJRNCHG syntax diagram
- Remove LAN Adapter (RMVLANADPT) command, RMVLANADPT syntax diagram
- Remove LAN Adapter Information (RMVLANADPI) command, RMVLANADPI syntax diagram
- Remove Library List Entry (RMVLIBLE) command, RMVLIBLE syntax diagram
- Remove License Key Information (RMVLICKEY) command, RMVLICKEY syntax diagram
- Remove Link (DEL) command, DEL syntax diagram
- Remove Link (ERASE) command, ERASE syntax diagram
- Remove Link (RMVLNK) command, RMVLNK syntax diagram
- Remove Log Entries from BRM (RMVLOGEBRM) command, RMVLOGEBRM syntax diagram
- Remove Log Entries from Job Scheduler (RMVLOGEJS) command, RMVLOGEJS syntax diagram
- Remove Media Information from BRM (RMVMEDIBRM) command, RMVMEDIBRM syntax diagram
- Remove Media Volumes from BRM (RMVMEDBRM) command, RMVMEDBRM syntax diagram
- Remove Member (RMVM) command, RMVM syntax diagram
- Remove Message (RMVMSG) command, RMVMSG syntax diagram
- Remove Message Description (RMVMSGD) command, RMVMSGD syntax diagram
- Remove Mounted File System (RMVMFS) command, RMVMFS syntax diagram
- Remove Mounted File System (UNMOUNT) command, UNMOUNT syntax diagram
- Remove NetWare Authentication Entry (RMVNTWAUTE) command, RMVNTWAUTE syntax diagram
- Remove Network Job Entry (RMVNETJOBE) command, RMVNETJOBE syntax diagram
- Remove Network Server Storage Link (RMVNWSSTGL) command, RMVNWSSTGL syntax diagram
- Remove Network Table Entry (RMVNETTBLE) command, RMVNETTBLE syntax diagram
- Remove Nickname (RMVNCK) command, RMVNCK syntax diagram
- Remove Node List Entry (RMVNODLE) command, RMVNODLE syntax diagram
- Remove Optical Cartridge (RMVOPTCTG) command, RMVOPTCTG syntax diagram
- Remove Optical Server (RMVOPTSVR) command, RMVOPTSVR syntax diagram
- Remove Performance Explorer Definition (RMVPEXDFN) command, RMVPEXDFN syntax diagram
- Remove Performance Explorer Filter (RMVPEXFTR) command, RMVPEXFTR syntax diagram
- Remove Physical File Constraint (RMVPFCST) command, RMVPFCST syntax diagram
- Remove Physical File Trigger (RMVPFTRG) command, RMVPFTRG syntax diagram
- Remove Point-to-Point TCP/IP Profile (RMVTCPTP) command, RMVTCPTP syntax diagram

- Remove Prestart Job Entry (RMVPJE) command, RMVPJE syntax diagram
- Remove Profile Tokens (RMVPRFTKN) command, RMVPRFTKN syntax diagram
- Remove Program (RMVPGM) command, RMVPGM syntax diagram
- Remove Program Temporary Fix (RMVPTF) command, RMVPTF syntax diagram
- Remove Protocol Table Entry (RMVPCLTBLE) command, RMVPCLTBLE syntax diagram
- Remove Relational Database Directory Entry (RMVRDBDIRE) command, RMVRDBDIRE syntax diagram
- Remove Remote Definition (RMVRMTDFN) command, RMVRMTDFN syntax diagram
- Remove Remote Journal (RMVRMTJRN) command, RMVRMTJRN syntax diagram
- Remove Remote Program Temporary Fix (RMVRMTPTF) command, RMVRMTPTF syntax diagram
- Remove Reply List Entry (RMVRPYLE) command, RMVRPYLE syntax diagram
- Remove REXX Buffer (RMVREXBUF) command, RMVREXBUF syntax diagram
- Remove Routing Entry (RMVRTGE) command, RMVRTGE syntax diagram
- Remove Search Index Entry (RMVSCHIDX) command, RMVSCHIDX syntax diagram
- Remove Server Authentication Entry (RMVSVRAUTE) command, RMVSVRAUTE syntax diagram
- Remove Service Table Entry (RMVSRVTBLE) command, RMVSRVTBLE syntax diagram
- Remove Sphere of Control Entry (RMVSOCE) command, RMVSOCE syntax diagram
- Remove Tape Cartridge (RMVTAPCTG) command, RMVTAPCTG syntax diagram
- Remove TCP/IP Host Table Entry (RMVTCPHTE) command, RMVTCPHTE syntax diagram
- Remove TCP/IP Interface (RMVTCPIFC) command, RMVTCPIFC syntax diagram
- Remove TCP/IP Port Restriction (RMVTCPPORT) command, RMVTCPPORT syntax diagram
- Remove TCP/IP Remote System Information (RMVTCPRSI) command, RMVTCPRSI syntax diagram
- Remove TCP/IP Route (RMVTCPRTE) command, RMVTCPRTE syntax diagram
- Remove TCP/IP Table (RMVTCPTBL) command, RMVTCPTBL syntax diagram
- Remove Trace (RMVTRC) command, RMVTRC syntax diagram
- Remove Trace Filter (RMVTRCFTR) command, RMVTRCFTR syntax diagram
- Remove Ultimedia System Facilities Connection Entry (RMVUSFCNNE) command, RMVUSFCNNE syntax diagram
- Remove Ultimedia System Facilities Device Entry (RMVUSFDEVE) command, RMVUSFDEVE syntax diagram
- Remove Ultimedia System Facilities Server Entry (RMVUSFSVRE) command, RMVUSFSVRE syntax diagram
- Remove Work Station Entry (RMVWSE) command, RMVWSE syntax diagram
- Rename (REN) command, REN syntax diagram
- Rename (RNM) command, RNM syntax diagram
- Rename Connection List Entry (RNMCNNLE) command, RNMCNNLE syntax diagram
- Rename Directory Entry (RNMDIRE) command, RNMDIRE syntax diagram
- Rename Diskette (RNMDKT) command, RNMDKT syntax diagram
- Rename Distribution List (RNMDSTL) command, RNMDSTL syntax diagram
- Rename Document Library Object (RNMDLO) command, RNMDLO syntax diagram
- Rename Job Using Job Scheduler (RNMJOBJS) command, RNMJOBJS syntax diagram
- Rename Local Area Network Adapter Information (RNMLANADPI) command, RNMLANADPI syntax diagram
- Rename Member (RNMM) command, RNMM syntax diagram
- Rename Nickname (RNMNCK) command, RNMNCK syntax diagram
- Rename Object (RNMOBJ) command, RNMOBJ syntax diagram

- Rename TCP/IP Host Table Entry (RNMTCPHTE) command, RNMTCPHTE syntax diagram
- Reorganize Document Library Object (RGZDLO) command, RGZDLO syntax diagram
- Reorganize Physical File Member (RGZPFM) command, RGZPFM syntax diagram
- Replace Document (RPLDOC) command, RPLDOC syntax diagram
- Request Order Assistance (RQSORDAST) command, RQSORDAST syntax diagram
- Reroute Job (RRTJOB) command, RRTJOB syntax diagram
- Restore (RST) command, RST syntax diagram
- Restore APAR Data (RSTAPARDTA) command, RSTAPARDTA syntax diagram
- Restore Authority (RSTAUT) command, RSTAUT syntax diagram
- Restore Authority Using BRM (RSTAUTBRM) command, RSTAUTBRM syntax diagram
- Restore Configuration (RSTCFG) command, RSTCFG syntax diagram
- Restore DLO Using BRM (RSTDLOBRM) command, RSTDLOBRM syntax diagram
- Restore Document Library Object (RSTDLO) command, RSTDLO syntax diagram
- Restore Library (RSTLIB) command, RSTLIB syntax diagram
- Restore Library Using BRM (RSTLIBBRM) command, RSTLIBBRM syntax diagram
- Restore Licensed Program (RSTLICPGM) command, RSTLICPGM syntax diagram
- Restore Object (RSTOBJ) command, RSTOBJ syntax diagram
- Restore Object Using BRM (RSTBRM) command, RSTBRM syntax diagram
- Restore Object Using BRM (RSTOBJBRM) command, RSTOBJBRM syntax diagram
- Restore System/36 File (RSTS36F) command, RSTS36F syntax diagram
- Restore System/36 Folder (RSTS36FLR) command, RSTS36FLR syntax diagram
- Restore System/36 Library Members (RSTS36LIBM) command, RSTS36LIBM syntax diagram
- Restore Ultimedia System Facilities Container (RSTUSFCNR) command, RSTUSFCNR syntax diagram
- Restore User Profiles (RSTUSRPRF) command, RSTUSRPRF syntax diagram
- Resume Breakpoint (RSMBKP) command, RSMBKP syntax diagram
- Resume Controller Recovery (RSMCTLRCY) command, RSMCTLRCY syntax diagram
- Resume Device Recovery (RSMDEVRCY) command, RSMDEVRCY syntax diagram
- Resume HTTP Crawl (RSMHTTPCRL) command
- Resume Line Recovery (RSMLINRCY) command, RSMLINRCY syntax diagram
- Resume Network Interface Recovery (RSMNWIRCY) command, RSMNWIRCY syntax diagram
- Resume Retrieve Using BRM (RSMRTVBRM) command, RSMRTVBRM syntax diagram
- Retrieve Authorization List Entry (RTVAUTLE) command, RTVAUTLE syntax diagram
- Retrieve Backup (RTVBCKUP) command, RTVBCKUP syntax diagram
- Retrieve Binder Source (RTVBNDSRC) command, RTVBNDSRC syntax diagram
- Retrieve CL Source (RTVCLSRC) command, RTVCLSRC syntax diagram
- Retrieve Cleanup (RTVCLNUP) command, RTVCLNUP syntax diagram
- Retrieve Configuration Source (RTVCFGSRC) command, RTVCFGSRC syntax diagram
- Retrieve Configuration Status (RTVCFGSTS) command, RTVCFGSTS syntax diagram
- Retrieve Current Directory (RTVCURDIR) command, RTVCURDIR syntax diagram
- Retrieve Data Area (RTVDTAARA) command, RTVDTAARA syntax diagram
- Retrieve Disk Information (RTVDSKINF) command, RTVDSKINF syntax diagram
- Retrieve Document (RTVDOC) command, RTVDOC syntax diagram
- Retrieve Document Library Object Authority (RTVDLOAUT) command, RTVDLOAUT syntax diagram
- Retrieve Document Library Object Name (RTVDLONAM) command, RTVDLONAM syntax diagram
- Retrieve Group Attributes (RTVGRPA) command, RTVGRPA syntax diagram

- Retrieve Job Attributes (RTVJOBA) command, RTVJOBA syntax diagram
- Retrieve Journal Entry (RTVJRNE) command, RTVJRNE syntax diagram
- Retrieve Library Description (RTVLIBD) command, RTVLIBD syntax diagram
- Retrieve Member Description (RTVMBRD) command, RTVMBRD syntax diagram
- Retrieve Message (RTVMSG) command, RTVMSG syntax diagram
- Retrieve Network Attributes (RTVNETA) command, RTVNETA syntax diagram
- Retrieve Object Description (RTVOBJD) command, RTVOBJD syntax diagram
- Retrieve Power On/Off Schedule Entry (RTVPWRSCDE) command, RTVPWRSCDE syntax diagram
- Retrieve Print Descriptor Group Profile (RTVPDGPRF) command, RTVPDGPRF syntax diagram
- Retrieve Product (RTVPRD) command, RTVPRD syntax diagram
- Retrieve Program Temporary Fix (RTVPTF) command, RTVPTF syntax diagram
- Retrieve Query Management Form (RTVQMFORM) command, RTVQMFORM syntax diagram
- Retrieve Query Management Query (RTVQMQR) command, RTVQMQR syntax diagram
- Retrieve Software Package Attributes (RTVSFWPKGA) command, RTVSFWPKGA syntax diagram
- Retrieve Stop Word List Source (RTVSWLSRC) command, RTVSWLSRC syntax diagram
- Retrieve System Information (RTVSYSINF) command, RTVSYSINF syntax diagram
- Retrieve System Manager Object (RTVSMGOBJ) command, RTVSMGOBJ syntax diagram
- Retrieve System Value (RTVSYSVAL) command, RTVSYSVAL syntax diagram
- Retrieve System/36 Attributes (RTVS36A) command, RTVS36A syntax diagram
- Retrieve Table Source (RTVTBSRC) command, RTVTBSRC syntax diagram
- Retrieve User Print Information (RTVUSRPRTI) command, RTVUSRPRTI syntax diagram
- Retrieve User Profile (RTVUSRPRF) command, RTVUSRPRF syntax diagram
- Retrieve Work Station Customizing Object Source (RTVWSCST) command, RTVWSCST syntax diagram
- Return (RETURN) command, RETURN syntax diagram
- Revoke Access Code Authority (RVKACCAUT) command, RVKACCAUT syntax diagram
- Revoke Object Authority (RVKOBJAUT) command, RVKOBJAUT syntax diagram
- Revoke Public Authority (RVKPUBAUT) command, RVKPUBAUT syntax diagram
- Revoke User Permission (RVKUSRPMN) command, RVKUSRPMN syntax diagram
- Revoke Workstation Object Authority (RVKWSOAUT) command, RVKWSOAUT syntax diagram
- Rollback (ROLLBACK) command, ROLLBACK syntax diagram
- Run Backup (RUNBCKUP) command, RUNBCKUP syntax diagram
- Run Java (RUNJVA) command, RUNJVA syntax diagram
- Run LPDA-2 (RUNLPDA) command, RUNLPDA syntax diagram
- Run Query (RUNQRY) command, RUNQRY syntax diagram
- Run Remote Command (AREXEC) command, AREXEC syntax diagram
- Run Remote Command (RUNRMTCMD) command, RUNRMTCMD syntax diagram
- Run Structured Query Language Statement (RUNSQLSTM) command, RUNSQLSTM syntax diagram
- Run System Manager Command (RUNSMGCMD) command, RUNSMGCMD syntax diagram
- Run System Manager Object (RUNSMGOBJ) command, RUNSMGOBJ syntax diagram

Back to the top

S

- Save (SAV) command, SAV syntax diagram
- Save APAR Data (SAVAPARDTA) command, SAVAPARDTA syntax diagram

- Save Changed Object (SAVCHGOBJ) command, SAVCHGOBJ syntax diagram
- Save Configuration (SAVCFG) command, SAVCFG syntax diagram
- Save DLO Using BRM (SAVDLOBRM) command, SAVDLOBRM syntax diagram
- Save Document Library Object (SAVDLO) command, SAVDLO syntax diagram
- Save Folder List Using BRM (SAVFLRLBRM) command, SAVFLRLBRM syntax diagram
- Save Library (SAVLIB) command, SAVLIB syntax diagram
- Save Library Using BRM (SAVLIBBRM) command, SAVLIBBRM syntax diagram
- Save Licensed Program (SAVLICPGM) command, SAVLICPGM syntax diagram
- Save Media Information Using BRM (SAVMEDIBRM) command, SAVMEDIBRM syntax diagram
- Save Object (SAVOBJ) command, SAVOBJ syntax diagram
- Save Object List Using BRM (SAVOBJLBRM) command, SAVOBJLBRM syntax diagram
- Save Object Using BRM (SAVBRM) command, SAVBRM syntax diagram
- Save Object Using BRM (SAVOBJBRM) command, SAVOBJBRM syntax diagram
- Save Save File Data (SAVSAVFDTA) command, SAVSAVFDTA syntax diagram
- Save Save Files Using BRM (SAVSAVFBRM) command, SAVSAVFBRM syntax diagram
- Save Security Data (SAVSECDDTA) command, SAVSECDDTA syntax diagram
- Save Storage (SAVSTG) command, SAVSTG syntax diagram
- Save System (SAVSYS) command, SAVSYS syntax diagram
- Save System Using BRM (SAVSYSBRM) command, SAVSYSBRM syntax diagram
- Save System/36 File (SAVS36F) command, SAVS36F syntax diagram
- Save System/36 Library Members (SAVS36LIBM) command, SAVS36LIBM syntax diagram
- Save Ultimeia System Facilities Container (SAVUSFCNR) command, SAVUSFCNR syntax diagram
- Save/Restore Changed Object (SAVRSTCHG) command, SAVRSTCHG syntax diagram
- Save/Restore Configuration (SAVRSTCFG) command, SAVRSTCFG syntax diagram
- Save/Restore Document Library Object (SAVRSTDLO) command, SAVRSTDLO syntax diagram
- Save/Restore Library (SAVRSTLIB) command, SAVRSTLIB syntax diagram
- Save/Restore Object (SAVRSTOBJ) command, SAVRSTOBJ syntax diagram
- Save/Restore Objects (SAVRST) command, SAVRST syntax diagram
- Select Command (SLTCMD) command, SLTCMD syntax diagram
- Send Break Message (SNDBRKMSG) command, SNDBRKMSG syntax diagram
- Send DBCS 3270PC Emulation Code (SNDEMLIGC) command, SNDEMLIGC syntax diagram
- Send Distribution (SNDDST) command, SNDDST syntax diagram
- Send Distribution Queue (SNDDSTQ) command, SNDDSTQ syntax diagram
- Send File (SNDF) command, SNDF syntax diagram
- Send Finance Diskette Image (SNDFNCIMG) command, SNDFNCIMG syntax diagram
- Send Journal Entry (SNDJRNE) command, SNDJRNE syntax diagram
- Send License (SNDLIC) command, SNDLIC syntax diagram
- Send Message (SNDMSG) command, SNDMSG syntax diagram
- Send Network File (SNDNETF) command, SNDNETF syntax diagram
- Send Network Message (SNDNETMSG) command, SNDNETMSG syntax diagram
- Send Network Spooled File (SNDNETSPLF) command, SNDNETSPLF syntax diagram
- Send Product (SNDPRD) command, SNDPRD syntax diagram
- Send Program Message (SNDPGMMSG) command, SNDPGMMSG syntax diagram
- Send Program Temporary Fix (SNDPTF) command, SNDPTF syntax diagram
- Send Program Temporary Fix Order (SNDPTFORD) command, SNDPTFORD syntax diagram

- Send Reply (SNDRPY) command, SNDRPY syntax diagram
- Send Reports Using Job Scheduler (SNDRPTJS) command, SNDRPTJS syntax diagram
- Send Service Request (SNDSRVRQS) command, SNDSRVRQS syntax diagram
- Send System Manager Object (SNDSMGOBJ) command, SNDSMGOBJ syntax diagram
- Send TCP/IP Spooled File (SNDTCPSPLF) command, SNDTCPSPLF syntax diagram
- Send Technical Information Exchange File (SNDTIEF) command, SNDTIEF syntax diagram
- Send User Message (SNDUSRMSG) command, SNDUSRMSG syntax diagram
- Send/Receive File (SNDRCVF) command, SNDRCVF syntax diagram
- Set ASP Group (SETASPGRP) command, SETASPGRP syntax diagram
- Set Attention Program (SETATNPGM) command, SETATNPGM syntax diagram
- Set Customization Data (SETCSTDTA) command, SETCSTDTA syntax diagram
- Set Dependent Job Using Job Scheduler (SETDEPJS) command, SETDEPJS syntax diagram
- Set Keyboard Map (SETKBDMAP) command, SETKBDMAP syntax diagram
- Set Media Controls Using BRM (SETMEDBRM) command, SETMEDBRM syntax diagram
- Set Object Access (SETOBJACC) command, SETOBJACC syntax diagram
- Set Retrieve Controls for BRM (SETRTVBRM) command, SETRTVBRM syntax diagram
- Set Tape Category (SETTAPCGY) command, SETTAPCGY syntax diagram
- Set User Usage for BRM (SETUSRBRM) command, SETUSRBRM syntax diagram
- Sign Off (SIGNOFF) command, SIGNOFF syntax diagram
- Start 3270 Display Emulation (STREML3270) command, STREML3270 syntax diagram
- Start Archive Using BRM (STRARCBRM) command, STRARCBRM syntax diagram
- Start ASP Balance (STRASPBAL) command, STRASPBAL syntax diagram
- Start Backup Using BRM (STRBKUBRM) command, STRBKUBRM syntax diagram
- Start Cleanup (STRCLNUP) command, STRCLNUP syntax diagram
- Start Cluster Node (STRCLUNOD) command, STRCLUNOD syntax diagram
- Start Cluster Resource Group (STRCRG) command, STRCRG syntax diagram
- Start Clustered Hash Table Server (STRCHTSVR) command, STRCHTSVR syntax diagram
- Start Commitment Control (STRCMTCTL) command, STRCMTCTL syntax diagram
- Start Communications Server (STRCMNSVR) command, STRCMNSVR syntax diagram
- Start Communications Trace (STRCMNTRC) command, STRCMNTRC syntax diagram
- Start Copy Screen (STRCPYSCN) command, STRCPYSCN syntax diagram
- Start Database Monitor (STRDBMON) command, STRDBMON syntax diagram
- Start Database Reader (STRDBRDR) command, STRDBRDR syntax diagram
- Start Debug (STRDBG) command, STRDBG syntax diagram
- Start Debug Server (STRDBGSVR) command, STRDBGSVR syntax diagram
- Start Directory Shadowing (STRDIRSHD) command, STRDIRSHD syntax diagram
- Start Disk Reorganization (STRDSKRGZ) command, STRDSKRGZ syntax diagram
- Start Diskette Reader (STRDKTRDR) command, STRDKTRDR syntax diagram
- Start Diskette Writer (STRDKTWTR) command, STRDKTWTR syntax diagram
- Start Education (STREDU) command, STREDU syntax diagram
- Start Expiration for BRM (STREXPBRM) command, STREXPBRM syntax diagram
- Start Font Management Aid (STRFMA) command, STRFMA syntax diagram
- Start Group Using Job Scheduler (STRGRPJS) command, STRGRPJS syntax diagram
- Start Host Server (STRHOSTSVR) command, STRHOSTSVR syntax diagram
- Start HTTP Crawl (STRHTTPCRL) command

- Start InfoSeeker (STRINFSKR) command, STRINFSKR syntax diagram
- Start Interactive Data Definition Utility (STRIDD) command, STRIDD syntax diagram
- Start Interactive Terminal Facility (STRITF) command, STRITF syntax diagram
- Start IP over SNA interface (STRIPSIFC) command, STRIPSIFC syntax diagram
- Start Job Scheduler (STRJS) command, STRJS syntax diagram
- Start Job Trace (STRJOBTRC) command, STRJOBTRC syntax diagram
- Start Journal (STRJRN) command, STRJRN syntax diagram
- Start Journal Access Path (STRJRNAP) command, STRJRNAP syntax diagram
- Start Journal Object (STRJRNOBJ) command, STRJRNOBJ syntax diagram
- Start Journal Physical File (STRJRNPf) command, STRJRNPf syntax diagram
- Start Mail Server Framework (STRMSF) command, STRMSF syntax diagram
- Start Maintenance for BRM (STRMNTBRM) command, STRMNTBRM syntax diagram
- Start Managed System Services (STRMGDSYS) command, STRMGDSYS syntax diagram
- Start Manager Services (STRMGRSRV) command, STRMGRSRV syntax diagram
- Start Migration Using BRM (STRMGRBRM) command, STRMGRBRM syntax diagram
- Start Mode (STRMOD) command, STRMOD syntax diagram
- Start Network File System Server (STRNFSSVR) command, STRNFSSVR syntax diagram
- Start Network Server Console (STRNWCSL) command, STRNWCSL syntax diagram
- Start Object Conversion (STROBJCVN) command, STROBJCVN syntax diagram
- Start Pass-Through (STRPASTHR) command, STRPASTHR syntax diagram
- Start Performance Explorer (STRPEX) command, STRPEX syntax diagram
- Start Performance Graphics (STRPFRG) command, STRPFRG syntax diagram
- Start Performance Tools (STRPFRT) command, STRPFRT syntax diagram
- Start Performance Trace (STRPFRTRC) command, STRPFRTRC syntax diagram
- Start Point-to-Point TCP/IP (STRTCPPTP) command, STRTCPPTP syntax diagram
- Start Prestart Jobs (STRPJ) command, STRPJ syntax diagram
- Start Printer Writer (STRPRTWTR) command, STRPRTWTR syntax diagram
- Start Program Profiling (STRPGMPRF) command, STRPGMPRF syntax diagram
- Start Programmer Menu (STRPGMMNU) command, STRPGMMNU syntax diagram
- Start QSH (STRQSH) command, STRQSH syntax diagram
- Start Query (STRQRY) command, STRQRY syntax diagram
- Start Query Management Procedure (STRQMPC) command, STRQMPC syntax diagram
- Start Query Management Query (STRQMQR) command, STRQMQR syntax diagram
- Start Question and Answer (STRQST) command, STRQST syntax diagram
- Start Recovery Using BRM (STRRCYBRM) command, STRRCYBRM syntax diagram
- Start Remote Support (STRRMTSPT) command, STRRMTSPT syntax diagram
- Start Remote Writer (STRRMTWTR) command, STRRMTWTR syntax diagram
- Start REXX Procedure (STRREXPRC) command, STRREXPRC syntax diagram
- Start RPC Binder Daemon (RPCBIND) command, RPCBIND syntax diagram
- Start Search Index (STRSCHIDX) command, STRSCHIDX syntax diagram
- Start Service Job (STRSRVJOB) command, STRSRVJOB syntax diagram
- Start Subsystem (STRSBS) command, STRSBS syntax diagram
- Start Subsystems Using BRM (STRSBSBRM) command, STRSBSBRM syntax diagram
- Start Support Network (STRSPTN) command, STRSPTN syntax diagram
- Start System Manager (STRSYSMGR) command, STRSYSMGR syntax diagram

- Start System Service Tools (STRSST) command, STRSST syntax diagram
- Start System/36 (STRS36) command, STRS36 syntax diagram
- Start System/36 Procedure (STRS36PRC) command, STRS36PRC syntax diagram
- Start TCP/IP (STRTCP) command, STRTCP syntax diagram
- Start TCP/IP Interface (STRTCPIFC) command, STRTCPIFC syntax diagram
- Start TCP/IP Server (STRTCPSVR) command, STRTCPSVR syntax diagram
- Start Technical Information Exchange Session (STRTISSN) command, STRTISSN syntax diagram
- Start Trace (STRTRC) command, STRTRC syntax diagram
- Start Trap Manager (STRTRPMGR) command, STRTRPMGR syntax diagram
- Start Ultimeia System Facilities (STRUSF) command, STRUSF syntax diagram
- Submit Change Request (SBMCRQ) command, SBMCRQ syntax diagram
- Submit Database Jobs (SBMDBJOB) command, SBMDBJOB syntax diagram
- Submit Diskette Jobs (SBMDKTJOB) command, SBMDKTJOB syntax diagram
- Submit Finance Job (SBMFNCJOB) command, SBMFNCJOB syntax diagram
- Submit Job (SBMJOB) command, SBMJOB syntax diagram
- Submit Job Using Job Scheduler (SBMJOBJS) command, SBMJOBJS syntax diagram
- Submit Network Job (SBMNETJOB) command, SBMNETJOB syntax diagram
- Submit Network Server Command (SBMNWSCMD) command, SBMNWSCMD syntax diagram
- Submit Remote Command (SBMRMTCMD) command, SBMRMTCMD syntax diagram

Back to the top

T

- Trace ASP Balance (TRCASPBAL) command, TRCASPBAL syntax diagram
- Trace Connection (TRCCNN) command, TRCCNN syntax diagram
- Trace CPI Communications (TRCCPIC) command, TRCCPIC syntax diagram
- Trace ICF (TRCICF) command, TRCICF syntax diagram
- Trace Internal (TRCINT) command, TRCINT syntax diagram
- Trace Job (TRCJOB) command, TRCJOB syntax diagram
- Trace REXX (TRCREX) command, TRCREX syntax diagram
- Trace Route (TRACROUTE) command, TRACROUTE syntax diagram
- Trace TCP/IP Application (TRCTCPAPP) command, TRCTCPAPP syntax diagram
- Trace TCP/IP Route (TRCTCPRTE) command, TRCTCPRTE syntax diagram
- Transfer Batch Job (TFRBCHJOB) command, TFRBCHJOB syntax diagram
- Transfer Control (TFRCTL) command, TFRCTL syntax diagram
- Transfer Job (TFRJOB) command, TFRJOB syntax diagram
- Transfer Pass-Through (TFRPASTHR) command, TFRPASTHR syntax diagram
- Transfer Secondary Job (TFRSECJOB) command, TFRSECJOB syntax diagram
- Transfer to Group Job (TFRGRPJOB) command, TFRGRPJOB syntax diagram

Back to the top

U

- Update Program (UPDPGM) command, UPDPGM syntax diagram
- Update Service Program (UPDSRVPGM) command, UPDSRVPGM syntax diagram
- Update System Information (UPDSYSINF) command, UPDSYSINF syntax diagram

[Back to the top](#)

V

- Vary Configuration (VRYCFG) command, VRYCFG syntax diagram
- Verify APPC Connection (APING) command, APING syntax diagram
- Verify APPC Connection (VFYAPPCCNN) command, VFYAPPCCNN syntax diagram
- Verify Communications (VFYCMN) command, VFYCMN syntax diagram
- Verify Image Catalog (VFYIMGCLG) command, VFYIMGCLG syntax diagram
- Verify Link Supporting LPDA-2 (VFYLNKLPDA) command, VFYLNKLPDA syntax diagram
- Verify Moves Using BRM (VFYMOVBRM) command, VFYMOVBRM syntax diagram
- Verify NetWare Authentication Entry (VFYNTWAUTE) command, VFYNTWAUTE syntax diagram
- Verify Optical (VFYOPT) command, VFYOPT syntax diagram
- Verify OptiConnect Connections (VFYOPCCNN) command, VFYOPCCNN syntax diagram
- Verify Printer (VFYPRT) command, VFYPRT syntax diagram
- Verify Tape (VFYTAP) command, VFYTAP syntax diagram
- Verify TCP/IP Connection (VFYTCCNN) command, VFYTCCNN syntax diagram

[Back to the top](#)

W

- Wait (WAIT) command, WAIT syntax diagram
- Work with Active Jobs (WRKACTJOB) command, WRKACTJOB syntax diagram
- Work with Alert Descriptions (WRKALRD) command, WRKALRD syntax diagram
- Work with Alert Tables (WRKALRTBL) command, WRKALRTBL syntax diagram
- Work with Alerts (WRKALR) command, WRKALR syntax diagram
- Work with APPN Status (WRKAPPNSTS) command, WRKAPPNSTS syntax diagram
- Work with ASP Descriptions (WRKASPB RM) command, WRKASPB RM syntax diagram
- Work with Authority (WRKAUT) command, WRKAUT syntax diagram
- Work with Authorization Lists (WRKAUTL) command, WRKAUTL syntax diagram
- Work with Binding Directories (WRKBNDDIR) command, WRKBNDDIR syntax diagram
- Work with Binding Directory Entries (WRKBNDDIRE) command, WRKBNDDIRE syntax diagram
- Work with Calendars Using BRM (WRKCALBRM) command, WRKCALBRM syntax diagram
- Work with Change Request Descriptions (WRKCRQD) command, WRKCRQD syntax diagram
- Work with Chart Formats (WRKCHTFMT) command, WRKCHTFMT syntax diagram
- Work with Class-of-Service Descriptions (WRKCOSD) command, WRKCOSD syntax diagram
- Work with Classes (WRKCLS) command, WRKCLS syntax diagram
- Work with Classes Using BRM (WRKCLSBRM) command, WRKCLSBRM syntax diagram
- Work with Commands (WRKCMD) command, WRKCMD syntax diagram
- Work with Commitment Definitions (WRKCMTDFN) command, WRKCMTDFN syntax diagram
- Work with Communications Side Information (WRKCSI) command, WRKCSI syntax diagram
- Work with Configuration Lists (WRKCFGL) command, WRKCFGL syntax diagram
- Work with Configuration Status (WRKCFGSTS) command, WRKCFGSTS syntax diagram
- Work with Connection List Entries (WRKCNNLE) command, WRKCNNLE syntax diagram
- Work with Connection Lists (WRKCNNL) command, WRKCNNL syntax diagram
- Work with Contact Information (WRKCNTINF) command, WRKCNTINF syntax diagram
- Work with Containers Using BRM (WRKCNRBRM) command, WRKCNRBRM syntax diagram

- Work with Control Groups Using BRM (WRKCTLGBRM) command, WRKCTLGBRM syntax diagram
- Work with Controller Descriptions (WRKCTLD) command, WRKCTLD syntax diagram
- Work with Data Areas (WRKDTAARA) command, WRKDTAARA syntax diagram
- Work with Data Definitions (WRKDTADFN) command, WRKDTADFN syntax diagram
- Work with Data Dictionaries (WRKDTADCT) command, WRKDTADCT syntax diagram
- Work with Data Queues (WRKDTAQ) command, WRKDTAQ syntax diagram
- Work with Database Files Using IDDU (WRKDBFIDD) command, WRKDBFIDD syntax diagram
- Work with Device Descriptions (WRKDEVD) command, WRKDEVD syntax diagram
- Work with Device Tables (WRKDEVTBL) command, WRKDEVTBL syntax diagram
- Work with Devices Using BRM (WRKDEVBRM) command, WRKDEVBRM syntax diagram
- Work with Directory Entries (WRKDIRE) command, WRKDIRE syntax diagram
- Work with Directory Locations (WRKDIRLOC) command, WRKDIRLOC syntax diagram
- Work with Directory Shadow Systems (WRKDIRSHD) command, WRKDIRSHD syntax diagram
- Work with Disk Status (WRKDSKSTS) command, WRKDSKSTS syntax diagram
- Work with Distributed Data Management Files (WRKDDMF) command, WRKDDMF syntax diagram
- Work with Distribution Catalog Entries (WRKDSTCLGE) command, WRKDSTCLGE syntax diagram
- Work with Distribution Lists (WRKDSTL) command, WRKDSTL syntax diagram
- Work with Distribution Queues (WRKDSTQ) command, WRKDSTQ syntax diagram
- Work with Document Libraries (WRKDOCLIB) command, WRKDOCLIB syntax diagram
- Work with Document Print Queue (WRKDOCPRTQ) command, WRKDOCPRTQ syntax diagram
- Work with Documents (WRKDOC) command, WRKDOC syntax diagram
- Work with DSNX/PC Distribution Queues (WRKDPCQ) command, WRKDPCQ syntax diagram
- Work with Edit Descriptions (WRKEDTD) command, WRKEDTD syntax diagram
- Work with Environment Variables (WRKENVVAR) command, WRKENVVAR syntax diagram
- Work with Files (WRKF) command, WRKF syntax diagram
- Work with Filter Action Entries (WRKFTRACNE) command, WRKFTRACNE syntax diagram
- Work with Filter Selection Entries (WRKFTRSLTE) command, WRKFTRSLTE syntax diagram
- Work with Filters (WRKFTR) command, WRKFTR syntax diagram
- Work with Folders (WRKFLR) command, WRKFLR syntax diagram
- Work with Font Resources (WRKFNTRSC) command, WRKFNTRSC syntax diagram
- Work with Form Definitions (WRKFORMDF) command, WRKFORMDF syntax diagram
- Work with Functional Areas (WRKFCNARA) command, WRKFCNARA syntax diagram
- Work with Graphics Symbol Sets (WRKGSS) command, WRKGSS syntax diagram
- Work with Hardware Resources (WRKHDWRSC) command, WRKHDWRSC syntax diagram
- Work with Held Optical Files (WRKHLDOPTF) command, WRKHLDOPTF syntax diagram
- Work with History Using Job Scheduler (WRKHSTJS) command, WRKHSTJS syntax diagram
- Work with HTTP Configuration (WRKHTTPCFG) command
- Work with Image Catalog Entries (WRKIMGCLGE) command, WRKIMGCLGE syntax diagram
- Work with IPX Descriptions (WRKIPXD) command, WRKIPXD syntax diagram
- Work with Job (WRKJOB) command, WRKJOB syntax diagram
- Work with Job Descriptions (WRKJOBDB) command, WRKJOBDB syntax diagram
- Work with Job Queues (WRKJOBQ) command, WRKJOBQ syntax diagram
- Work with Job Schedule Entries (WRKJOBSCDE) command, WRKJOBSCDE syntax diagram
- Work with Jobs Using Job Scheduler (WRKJOBJS) command, WRKJOBJS syntax diagram
- Work with Journal (WRKJRN) command, WRKJRN syntax diagram

- Work with Journal Attributes (WRKJRNA) command, WRKJRNA syntax diagram
- Work with Journal Receivers (WRKJRNRCV) command, WRKJRNRCV syntax diagram
- Work with LAN Adapters (WRKLANADPT) command, WRKLANADPT syntax diagram
- Work with Libraries (WRKLIB) command, WRKLIB syntax diagram
- Work with License Information (WRKLICINF) command, WRKLICINF syntax diagram
- Work with Line Descriptions (WRKLIND) command, WRKLIND syntax diagram
- Work with Link Information (WRKLNKBRM) command, WRKLNKBRM syntax diagram
- Work with Lists Using BRM (WRKLBRM) command, WRKLBRM syntax diagram
- Work with Locations Using BRM (WRKLOCBRM) command, WRKLOCBRM syntax diagram
- Work with Media Information (WRKMEDIBRM) command, WRKMEDIBRM syntax diagram
- Work with Media Libraries Using BRM (WRKMLBBRM) command, WRKMLBBRM syntax diagram
- Work with Media Library Media (WRKMLMBRM) command, WRKMLMBRM syntax diagram
- Work with Media Library Resource Queue (WRKMLBRSCQ) command, WRKMLBRSCQ syntax diagram
- Work with Media Library Status (WRKMLBSTS) command, WRKMLBSTS syntax diagram
- Work with Media Using BRM (WRKMEDBRM) command, WRKMEDBRM syntax diagram
- Work with Menus (WRKMNU) command, WRKMNU syntax diagram
- Work with Message Descriptions (WRKMSGD) command, WRKMSGD syntax diagram
- Work with Message Files (WRKMSGF) command, WRKMSGF syntax diagram
- Work with Message Queues (WRKMSGQ) command, WRKMSGQ syntax diagram
- Work with Messages (WRKMSG) command, WRKMSG syntax diagram
- Work with Migration Information Using BRM (WRKMGRIBRM) command, WRKMGRIBRM syntax diagram
- Work with Mode Descriptions (WRKMODD) command, WRKMODD syntax diagram
- Work with Modules (WRKMOD) command, WRKMOD syntax diagram
- Work with NetBIOS Descriptions (WRKNTBD) command, WRKNTBD syntax diagram
- Work with NetWare Authentication Engr (WRKNTWAUTE) command, WRKNTWAUTE syntax diagram
- Work with NetWare Volumes (WRKNTWVOL) command, WRKNTWVOL syntax diagram
- Work with Network Files (WRKNETF) command, WRKNETF syntax diagram
- Work with Network Interface Description (WRKNWID) command, WRKNWID syntax diagram
- Work with Network Job Entries (WRKNETJOB) command, WRKNETJOB syntax diagram
- Work with Network Server Descriptions (WRKNWSD) command, WRKNWSD syntax diagram
- Work with Network Server Status (WRKNWSSTS) command, WRKNWSSTS syntax diagram
- Work with Network Server Storage Spaces (WRKNWSSTG) command, WRKNWSSTG syntax diagram
- Work with Network Server User Enrollment (WRKNWSEN) command, WRKNWSEN syntax diagram
- Work with Network Table Entry (WRKNETTBLE) command, WRKNETTBLE syntax diagram
- Work with Nicknames (WRKNCK) command, WRKNCK syntax diagram
- Work with Node List Entries (WRKNODLE) command, WRKNODLE syntax diagram
- Work with Node Lists (WRKNODL) command, WRKNODL syntax diagram
- Work with Object Links (WRKLNK) command, WRKLNK syntax diagram
- Work with Object Locks (WRKOBJLCK) command, WRKOBJLCK syntax diagram
- Work with Objects (WRKOBJ) command, WRKOBJ syntax diagram
- Work with Objects by Owner (WRKOBJOWN) command, WRKOBJOWN syntax diagram
- Work with Objects by Primary Group (WRKOBJPGP) command, WRKOBJPGP syntax diagram
- Work with Optical Directories (WRKOPTDIR) command, WRKOPTDIR syntax diagram

- Work with Optical Files (WRKOPTF) command, WRKOPTF syntax diagram
- Work with Optical Volumes (WRKOPTVOL) command, WRKOPTVOL syntax diagram
- Work with OptiConnect Activity (WRKOPCACT) command, WRKOPCACT syntax diagram
- Work with Order Information (WRKORDINF) command, WRKORDINF syntax diagram
- Work with Order Requests (WRKORDRQS) command, WRKORDRQS syntax diagram
- Work with Output Queue Description (WRKOUTQD) command, WRKOUTQD syntax diagram
- Work with Output Queues (WRKOUTQ) command, WRKOUTQ syntax diagram
- Work with Overlays (WRKOVL) command, WRKOVL syntax diagram
- Work with Page Definitions (WRKPAGDFN) command, WRKPAGDFN syntax diagram
- Work with Page Segments (WRKPAGSEG) command, WRKPAGSEG syntax diagram
- Work with Panel Groups (WRKPNLGRP) command, WRKPNLGRP syntax diagram
- Work with Performance Explorer Definition (WRKPEXDFN) command, WRKPEXDFN syntax diagram
- Work with Performance Explorer Filter (WRKPEXFTR) command, WRKPEXFTR syntax diagram
- Work with Physical File Constraints (WRKPFCST) command, WRKPFCST syntax diagram
- Work with Physical File Datalinks (WRKPFDL) command, WRKPFDL syntax diagram
- Work with Point-to-Point TCP/IP (WRKTCPPTP) command, WRKTCPPTP syntax diagram
- Work with Policies Using BRM (WRKPCYBRM) command, WRKPCYBRM syntax diagram
- Work with Print Services Facility Configurations (WRKPSFCFG) command, WRKPSFCFG syntax diagram
- Work with Printing Status (WRKPRTSTS) command, WRKPRTSTS syntax diagram
- Work with Problems (WRKPRB) command, WRKPRB syntax diagram
- Work with Product Information (WRKPRDINF) command, WRKPRDINF syntax diagram
- Work with Program Tables (WRKPGMTBL) command, WRKPGMTBL syntax diagram
- Work with Programs (WRKPGM) command, WRKPGM syntax diagram
- Work with Protocol Table Entry (WRKPCLTBLE) command, WRKPCLTBLE syntax diagram
- Work with PTF (WRKPTF) command, WRKPTF syntax diagram
- Work with PTF Groups (WRKPTFGRP) command, WRKPTFGRP syntax diagram
- Work with Query Management Forms (WRKQMFORM) command, WRKQMFORM syntax diagram
- Work with Query Management Queries (WRKQMQRY) command, WRKQMQRY syntax diagram
- Work with Questions (WRKQST) command, WRKQST syntax diagram
- Work with Readers (WRKRDR) command, WRKRDR syntax diagram
- Work with Received Change Request Activities (WRKRCVCRQA) command, WRKRCVCRQA syntax diagram
- Work with Recovery Activities (WRKRCYBRM) command, WRKRCYBRM syntax diagram
- Work with Registration Information (WRKREGINF) command, WRKREGINF syntax diagram
- Work with Relational Database Directory Entries (WRKRDBDIRE) command, WRKRDBDIRE syntax diagram
- Work with Remote Definitions (WRKRMTDFN) command, WRKRMTDFN syntax diagram
- Work with Routed Configuration (WRKRTDCFG) command, WRKRTDCFG syntax diagram
- Work with Save Files Using BRM (WRKSAVFBRM) command, WRKSAVFBRM syntax diagram
- Work with Saved Folders Using BRM (WRKFLRBRM) command, WRKFLRBRM syntax diagram
- Work with Saved Objects Using BRM (WRKOBJBRM) command, WRKOBJBRM syntax diagram
- Work with Saved Spooled Files (WRKSPLFBRM) command, WRKSPLFBRM syntax diagram
- Work with Search Index Entries (WRKSCHIDX) command, WRKSCHIDX syntax diagram
- Work with Search Indexes (WRKSCHIDX) command, WRKSCHIDX syntax diagram

- Work with Service Programs (WRKSRVPGM) command, WRKSRVPGM syntax diagram
- Work with Service Providers (WRKSRVPVD) command, WRKSRVPVD syntax diagram
- Work with Service Requesters (WRKSRVRQS) command, WRKSRVRQS syntax diagram
- Work with Service Table Entry (WRKSRVTBLE) command, WRKSRVTBLE syntax diagram
- Work with Shared Storage Pools (WRKSHRPOOL) command, WRKSHRPOOL syntax diagram
- Work with Software Agreements (WRKSFWAGR) command, WRKSFWAGR syntax diagram
- Work with Spelling Aid Dictionaries (WRKSPADCT) command, WRKSPADCT syntax diagram
- Work with Sphere of Control (WRKSOC) command, WRKSOC syntax diagram
- Work with Spooled File Attributes (WRKSPLFA) command, WRKSPLFA syntax diagram
- Work with Spooled Files (WRKSPLF) command, WRKSPLF syntax diagram
- Work with Submitted Change Requests (WRKSBMCRQ) command, WRKSBMCRQ syntax diagram
- Work with Submitted Change Requests Activities (WRKSBMCRQA) command, WRKSBMCRQA syntax diagram
- Work with Submitted Jobs (WRKSBMJOB) command, WRKSBMJOB syntax diagram
- Work with Subsystem Descriptions (WRKSBSD) command, WRKSBSD syntax diagram
- Work with Subsystem Jobs (WRKSBSJOB) command, WRKSBSJOB syntax diagram
- Work with Subsystems (WRKSBS) command, WRKSBS syntax diagram
- Work with Supported Products (WRKSPTPRD) command, WRKSPTPRD syntax diagram
- Work with System Activity (WRKSYSACT) command, WRKSYSACT syntax diagram
- Work with System Reply List Entries (WRKRPYLE) command, WRKRPYLE syntax diagram
- Work with System Status (WRKSYSSTS) command, WRKSYSSTS syntax diagram
- Work with System Values (WRKSYSVAL) command, WRKSYSVAL syntax diagram
- Work with System/36 (WRKS36) command, WRKS36 syntax diagram
- Work with System/36 Procedure Attributes (WRKS36PRCA) command, WRKS36PRCA syntax diagram
- Work with System/36 Program Attributes (WRKS36PGMA) command, WRKS36PGMA syntax diagram
- Work with System/36 Source Attributes (WRKS36SRCA) command, WRKS36SRCA syntax diagram
- Work with Tables (WRKTBL) command, WRKTBL syntax diagram
- Work with Tape Cartridges (WRKTAPCTG) command, WRKTAPCTG syntax diagram
- Work with TCP/IP Network Status (WRKTCPSTS) command, WRKTCPSTS syntax diagram
- Work with Technical Information Exchange (WRKTIE) command, WRKTIE syntax diagram
- Work with Ultimedia System Facilities Connection Entries (WRKUSFCNNE) command, WRKUSFCNNE syntax diagram
- Work with Ultimedia System Facilities Device Entries (WRKUSFDEVE) command, WRKUSFDEVE syntax diagram
- Work with Ultimedia System Facilities Server Entries (WRKUSFSVRE) command, WRKUSFSVRE syntax diagram
- Work with User Jobs (WRKUSRJOB) command, WRKUSRJOB syntax diagram
- Work with User Profiles (WRKUSRPRF) command, WRKUSRPRF syntax diagram
- Work with User Tables (WRKUSRTBL) command, WRKUSRTBL syntax diagram
- Work with Writers (WRKWTR) command, WRKWTR syntax diagram

CL concepts and reference

This section describes concepts and provides reference information you may need when using Control Language (CL) commands.

- “Commands” on page 45
- “OS/400 objects” on page 137

- “Commonly used parameters: Expanded descriptions” on page 144
- “Database and device files used by CL commands” on page 173
- “Printing command descriptions on the server” on page 199

Commands

You can use Control Language (CL) commands to request system functions. See the following for command concepts.

- “Command names”
- “Commands operating on OS/400 objects”
- “Commands operating on multiple objects” on page 46
- “Command description format” on page 46
- “Syntax diagram format” on page 47
- “Command parts” on page 57
- “CL programs” on page 71
- “Command definition statements” on page 87
- “Parameter values used for testing and debugging” on page 134

Command names

Most command names consist of a combination of the verb and the object being acted on: (command = verb + object acted on). For example, you can create, delete, or display a library; so the verb abbreviations CRT, DLT, and DSP are joined to the abbreviation for library, LIB. The result is three commands that can operate on a library: CRTLIB, DLTLIB, and DSPLIB.

The conventions for naming the combination verb and object commands are as follows:

- The primary convention (as just shown) is to use three letters from each word in the descriptive command name to form the abbreviated command name that is recognized by the system.
- The secondary convention is to use single letters from the ending word or words in the command title for the end of the command name, such as the three single letters DLO on the DLTDLLO (Delete Document Library Object) command.
- An occasional convention is to use single letters in the middle of the command name (usually between commonly used three-character verbs and objects), such as the letters CL in the CRTCLPGM (Create CL Program) command.

Some command names consist of the verb only, such as the MOV (Move) command, or an object only, such as the DATA (Data) command.

For a complete list of all abbreviations used in command (and keyword) names, see the Object Types Used by Commands Containing the OBJTYPE Parameter (“Table 8. Object Types Used by Commands Containing the OBJTYPE Parameter” on page 160) table.

A few commands have an OS/400 command name, and can also be called using one or more alternate names that may be familiar to users of systems other than the OS/400 system. An alternate name is known as an **alias**, such as the name CD is an alias for the CHGCURDIR (Change Current Directory) command. Aliases are presented in this topic with either the complete command description of the OS/400 command, or with a reference to the command name that presents the complete command description.

Commands operating on OS/400 objects

Each of the OS/400 object types has a set of commands that operates on that object type. Most OS/400 object types have commands that perform the following actions:

- Create (CRT): Creates the object and specifies its attributes.

- Delete (DLT): Deletes the object from the system.
- Change (CHG): Changes the attributes and/or contents of the object.
- Display (DSP): Displays the contents of the object. Display commands cannot be used to operate on objects.
- Work with (WRK): Works with the attributes and/or contents of the object. Unlike display commands, work commands allow users to operate on objects and modify applications.

For additional information, see “Commands operating on multiple objects”.

Commands operating on multiple objects

In addition to the commands that operate on single object types, there are commands that operate on several object types. These commands are more powerful because they can operate on several objects of different types at the same time. For example:

- Display object description (DSPOBJD): Displays the common attributes of an object.
- Move object (MOV OBJ): Moves an object from one library to another.
- Rename object (RNMOBJ): Specifies the new name of an object.
- Save object (SAVOBJ): Saves an object and its contents on diskette, tape, optical media, or in a save file.
- Restore object (RSTOBJ): Restores a saved version of the object from diskette, tape, optical media, or from a save file.

The following table shows the commands that perform an action on many of the object types. Some of the commands, such as the Move Object (MOV OBJ) command, operate on only one object at a time, but that object can be any one of several OS/400 object types. For example, the MOV OBJ command can move a file or a job description.

Table 1. Commands Operating on Multiple Object Types

Item	Identifier	Actions
Object	OBJ	ALC, CHK, CPR, DCP, DLC, DMP, MOV, RNM, RST, SAV, WRK, CRTDUP, SAVCHG
Object Access	OBJACC	SET
Object Auditing	OBJAUD	CHG
Object Authority	OBJAUT	DSP, EDT, GRT, RVK
Object Description	OBJD	DSP
Object Integrity	OBJITG	CHK
Object Lock	OBJLCK	WRK
Object Owner	OBJOWN	CHG, WRK
Object Primary Group	OBJPGP	CHG, WRK


You can also refer to the Object Types Used by Commands Containing the OBJTYPE Parameter (“Table 8. Object Types Used by Commands Containing the OBJTYPE Parameter” on page 160) table in “OBJTYPE parameter” on page 159 to see how these multiple-object commands affect specific object types.

For additional information, see “Commands operating on OS/400 objects” on page 45.

Command description format

Each command description follows the same format and includes the parts discussed in the following paragraphs.

It should be noted that, because a command is an iSeries 400 object, each command can be authorized for specific users or authorized for use by the public (all users authorized in some way to use the system). Because this is true for nearly every command, it is not stated in each command description. The iSeries

Security Reference  book, contains additional information about IBM-supplied user profiles and the commands authorized for each.

Command syntax

The command syntax is presented in the syntax diagram. The syntax diagram shows all the parameters and values that are valid for the command. The parameters are divided into two groups: those that must be coded (required), and those that need not be coded (optional). Default values are indicated by their position above the base line. These default values are used by the system for parameters or parts of parameters that are not coded.

The “Syntax diagram format” topic provides an detailed, in-depth discussion of command syntax.

Command description

The general description of the command (under the heading “Purpose”) briefly explains the function of the command and any relationships it has with a program or with other commands. If there are restrictions on the use of the command, they are described under the heading “Restrictions.”

Parameter descriptions

Parameter descriptions in the text are presented in the same order as in the syntax diagram. The syntax diagram shows the order in which the parameters must be specified if the values are specified in positional form (that is, without keywords). If a parameter has more than one value, the values are described in the same order as shown in the syntax diagram. The default value, if there is one, is always first and is shown as an underlined heading at the beginning of the text that describes the value.

The description of each parameter explains what the parameter means, what it specifies, and the dependent relationships it has with other parameters in the command. When the parameter has more than one value, the information that applies to the parameter as a whole is covered first, then the specific information for each of the values is described after the name of each value.

Command coding examples

Each command description with parameters shows at least one coded example. Where necessary, several examples are provided for commands with many parameters and several logical combinations.

For clarity, each example is coded in keyword form only. The same examples could be coded either in positional form or in a combination of keyword and positional forms.

Additional command considerations

A section called “Additional Considerations” follows the coded examples of some commands to present additional useful information about the command.

Syntax diagram format

Syntax diagrams for each command show how you enter the command and its parameters and values in programs, in batch, or interactively on the command line.

See the following topics for information about syntax diagram format.

- “How to read syntax diagrams” briefly describes the basic rules you need to read syntax diagrams.
- “How syntax diagrams show command structure” on page 49 describes in more detail how syntax diagrams reflect CL command structure.
- “Syntax diagram rules” on page 49 provides both the rules, along with a graphic depiction, for syntax diagrams.

To view the syntax diagram for a specific command, see the Alphabetic list of commands or the “CL command finder” on page 3 topic(s) in the navigation bar.

How to read syntax diagrams

Syntax diagrams show the parameters and values used by each CL command. Each syntax diagram specifies, for one command, the parameters that can be coded in the command and the choice of values that are valid for each parameter. Refer to “How syntax diagrams show command structure” on page 49 for a more detailed description of syntax diagrams and how they reflect command structure. The following list describes the individual pieces of a syntax diagram and how to read them:

- *Start here* >>— and end here —<<. This line with the starting and ending arrows is considered the syntax base line.
- When a *parameter keyword* is entered, its **predefined value** (values beginning with * or Q), or **user-defined value** must be entered with parentheses as shown below:

```
>>-KEYWORD(-----+*VALUE1-----+---)-----<<
                +-*VALUE2-----+
                '-user-defined-value-'
```

- *Required parameters* must be specified. They are presented with the parameter keyword on the base line.

Optional parameters do not have to be entered. They are represented with the parameter keyword below the base line.

In this sample, you must enter KEYWORD1 with a value, but you do not have to enter KEYWORD2.

```
>>-KEYWORD1(--+*VALUE1+----)----->
                '-*VALUE2-'
```

```
>-----+-----+-----<
                '-KEYWORD2(--+*VALUE1----)---'
                '-any-value-'
```

- *Default values* do not have to be entered. They are used when you do not specify a parameter. In this sample, you can enter *VALUE1, *VALUE2, or nothing. If you enter nothing, *VALUE1 is assumed.

```
>>+-----+-----<
    |           .-*VALUE1--.   |
    '-KEYWORD(--+*VALUE2--+----)---'
```

- *Repeated values* can be specified for some parameters. In this sample, the arrow moving from the right of the values, over the top of the values, and down the left of the values to the arrowhead, indicates that the values inside the path of the arrow can be repeated. The accompanying footnote indicates the maximum number of times that values can be specified (10 times).

```

                .-----
                v     .-*VAL1--.   .-*VAL3--.   (1) |
>>-KEYWORD(-----(-+*VAL2--+-----+-----)-----<<
                '-value--'
```

Note:

1. A maximum of 10 repetitions can be specified.
- *Fragments* allow you to label entire blocks of parameters or values and describe them in another section of the syntax diagram.

```
>>-KEYWORD(-- | Fragment | ---)-----<<
```

Fragment

```

    .-*VALUE1--.
|---+-----+---+VALUE3-----+-----|
    '-*VALUE2--'  '+-VALUE4-----+'
                    '-other-values-'

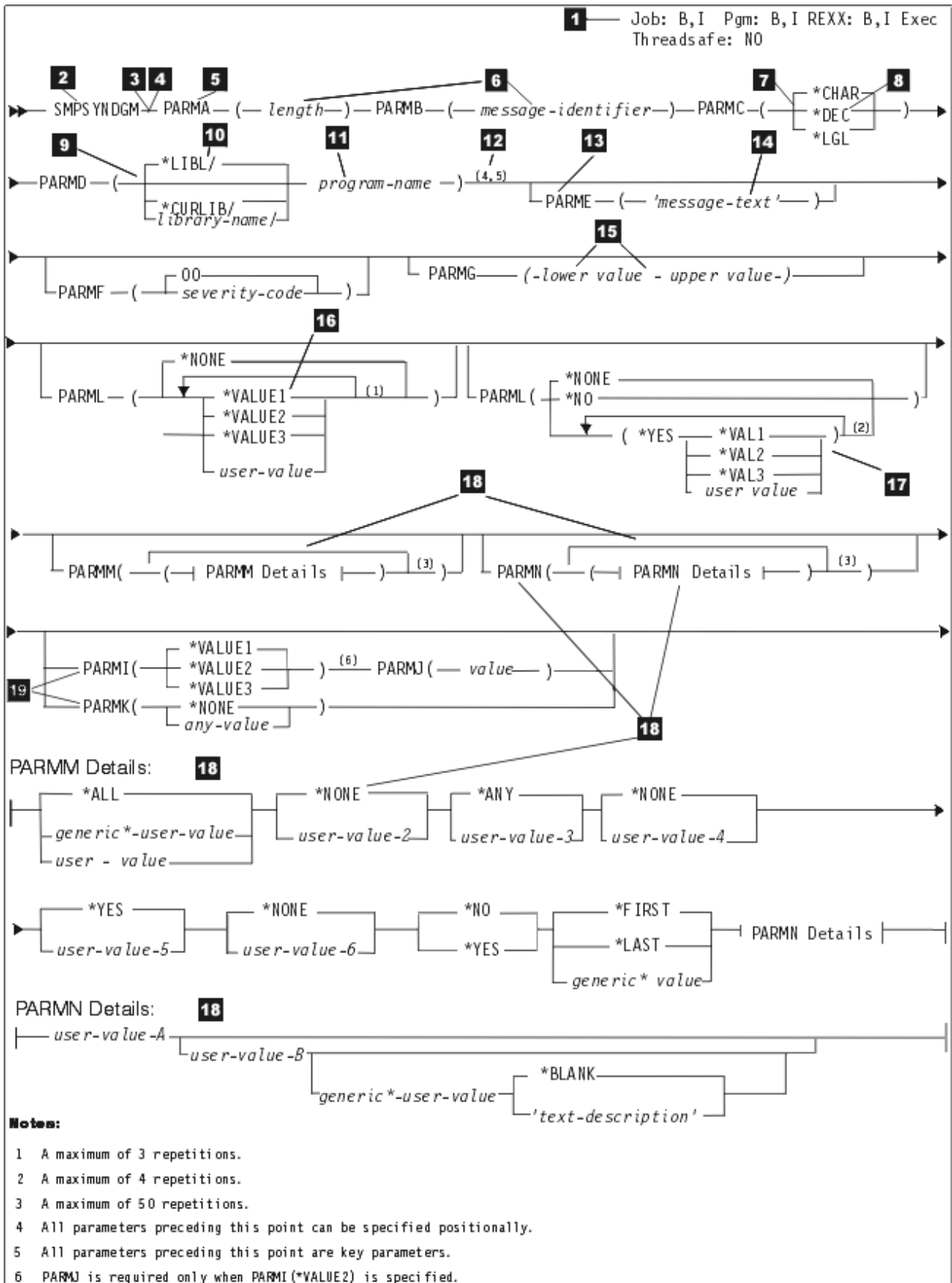
```

How syntax diagrams show command structure

In addition to showing simple command structures such as parameter keywords and values, syntax diagrams also show more complicated command structures such as mutually exclusive parameters. The “Syntax diagram rules” section describes in more detail how the syntax diagrams show these command structures, and it shows the parameter syntax of a fictitious command named Sample Syntax Diagram (SMPSYNDGM).

Syntax diagram rules

This sample syntax diagram shows several parameter structures and how those structures are represented on the syntax diagram. In addition, the sample syntax diagram contains additional information about the command, its parameters, and its values (such as where the command can be used, or whether it is threadsafe, see **(1)**). The rules on how to interpret this diagram follow this syntax diagram



The syntax diagrams for the CL commands are interpreted according to the following rules.

(1) Entry Codes (Batch and Interactive)

The box insert in the upper right corner of each syntax diagram contains the entry codes that specify the environment in which the command can be entered. The codes indicate whether the command can be:

- Used in a job (outside a compiled program; Job:B and/or I). When used in this manner, the command is considered a separate entity inside the job, and is run by itself as a separate function in what is called the interpretive mode. That is, commands in batch mode and/or interactive jobs that are not in compiled programs are interpreted and run one at a time, one after the other. The function of one interpreted command in the job is completed before the next command is interpreted.
- Used in a compiled program (Pgm:B and/or I). In this case, the command is part of the program; the command is in compiled form with the rest of the program, and running the command depends upon when the program is called and upon the program's logic preceding the command. That is, a compiled command cannot be run unless the program it is in is run.
- Used in a CL ILE module only (Mod:B and/or I).
- Used in a REXX procedure (REXX:B and/or I). In this case, the command is part of the REXX procedure. Running the command depends upon when the REXX procedure is started with the Start REXX Procedure (STRREXPRC) command. That is, the command cannot be run unless the procedure containing it is started.

The explanations of the combinations of entry codes are shown in the following table:

Table 9. Entry Codes (Batch and Interactive)

Code	Representing	Meaning
Job:B	Batch job	Valid in batch input stream, external to compiled CL programs.
Job:I	Interactive job	Valid for interactive entry, external to compiled CL programs.
Job:B,I	Batch and interactive jobs	Valid for batch and interactive entry, external to compiled CL programs.
Pgm:B	Program, batch	Valid in a compiled CL program that is called from batch entry.
Pgm:I	Program, interactive	Valid in a compiled CL program that is called from interactive entry.
Pgm:B,I	Program, batch and interactive	Valid in a compiled CL program that is called from batch or interactive entry.
Mod:B	Module, batch	Valid in a batch CL ILE module only.
Mod:I	Module, interactive	Valid in an interactive CL ILE module only.
Mod:B,I	Module, batch and interactive	Valid in a CL ILE module that is called from batch or interactive entry.
REXX:B	REXX, batch	Valid in a REXX procedure that is called from batch entry.
REXX:I	REXX, interactive	Valid in a REXX procedure that is called from interactive entry.
REXX:B,I	REXX, batch and interactive	Valid in a REXX procedure that is called from batch or interactive entry.
Exec	QCPCMD and QCMDXC processing	Valid as a parameter on the CALL command; can be passed as a character string to the system programs QCPCMD and QCMDXC.

Threadsafe (Yes, Conditional, No)

The box insert in the upper right corner of each syntax also indicates whether a command is threadsafe. Each command in this book has a threadsafe classification. The three types of threadsafe classifications are as follows:

- **Threadsafe: Yes**

This classification indicates that you can safely call the command simultaneously in multiple threads without restrictions. This classification also indicates that all functions called by this command are threadsafe.

- **Threadsafe: Conditional**

This classification indicates that not all functions provided by the command are threadsafe. The Restrictions section of the command provides information relating to thread safety limitations. Many commands are classified conditionally threadsafe because either some underlying system support is not threadsafe or the command can cause an exit point to be called. Some conditionally threadsafe commands may deny access under some circumstances. The command restriction section describes the conditions that cause the command to deny access.

- **Threadsafe: No**

This classification indicates that the command is not threadsafe and should not be used in a multithreaded program. While some thread unsafe commands may deny access, most thread unsafe commands do not. A diagnostic message, CPD000D, may be sent to the job log to indicate that a non-threadsafe command has been called. Whether or not the diagnostic message CPD000D is sent to the job log depends on the “multithreaded job action” attribute of the command; that attribute can be determined by using the Display Command (DSPCMD) command. The possible values and actions are:

- *SYSVAL - Action is based on system value QMLTTHDACN
- *RUN - Command runs. No messages are sent.
- *MSG - Diagnostic message CPD000D is sent to the job log. The command runs.
- *NORUN - Diagnostic message CPD000D is sent to the job log, and escape message CPF0001 is sent. The command does not run.

If the command is run, the results are unpredictable.

(2) *Command Name*

The command name appears first in the diagram. In the sample syntax diagram, the name of the fictitious command is SMPSYNDGM.

(3) *Command Labels*

The labels that may precede all coded commands are optional. Therefore, labels are not shown in the syntax diagrams. If used, a label must be followed by a colon (:) immediately after the last character in the label name.

(4) *Parameter Order*

All parameters of the command are shown in the correct positional sequence up to the positional coding limit (see rule 12). The order goes left to right on each line and continues on the following line. To show the positional order of the parameters in the sample diagram, the representative parameter keyword names are named in alphabetic order, such as PARMA and PARMB. They are named in the same order in which they would have to be coded in positional form if this command actually existed.

Note: In the few cases where dependent parameter relationships are shown (see rule (19)), the positional parameter order may not be readily apparent. The parameter order may be specified in a note in the diagram, or it can be easily determined from the text where all parameters in the command are described in positional order.

When coding parameters in positional form, you must enter them in the order shown in the diagram. If you choose not to code a parameter and another positional parameter is coded after it, then you must enter *N to represent the uncoded parameter in the positional sequence.

Parentheses must be coded in each parameter that either has the keyword coded with its value (see rule (5)) or has multiple values in one parameter (see rule (17)).

(5) Parameter Keywords

For each parameter, the keyword is always shown first, followed by the parameter values. Parameter keywords use uppercase letters; if you code them in lowercase, they will be changed to uppercase.

When a parameter is coded in keyword form, its associated values must be enclosed in parentheses as shown below:

```
>>-PARMA(---length---)-----<
```

This parameter is coded as follows:

```
PARMA(15)
```

(6) User-Defined Values

User-defined values are shown with lowercase characters that describe the kind of value coded by the user. If more than one word is used to describe a single value, the words are connected by hyphens:

```
>>-PARMB(---message-identifier---)-----<
```

(7) Choice of Values

For a parameter having a choice of values, only one of which can be specified (typed after the parameter keyword and enclosed in parentheses), the choices of values are shown on different *branch* lines that follow the parameter keyword (which is on the *base* line), as follows:

```
>>-PARMC(----+-*CHAR+---)-----<
      +-*DEC--+
      '-*LGL--'
```

(8) Predefined Values

Predefined values are shown exactly as they must be coded. Predefined values usually begin with an * followed by all uppercase letters. *CHAR and *DEC are examples of predefined values. Predefined values can also begin with the letter Q. For example, an option for the SRCFILE parameter on the CRTCLPGM command is QCLSRC.

(9) Optional Values

Optional values are shown by a blank line as shown below:

```
      .-*LIBL/-----
>>-PARMD(----+-----+---program---)-----<
      +-*CURLIB/-----+
      '-library-name/-'
```

The blank line indicates that a value from the first group (which includes *LIBL, *CURLIB, and *library-name*) does not have to be entered as shown below:

```
PARMD(CLPGM)
```

If the blank line were missing, you would have to code the parameter as shown below:

PARMD(*LIBL/CLPGM) or
 PARMD(*CURLIB/CLPGM) or
 PARMD(MYLIB/CLPGM)

(10) Default Values

If a parameter has a default value, it is indicated by its position above the base line. The value *LIBL (see (9)), is a default library qualifier on the PARMD parameter, and 00 is the default value on the PARMF parameter as shown below:

```
>>+-----+-----<<
|          .-00-----|
|'-PARMF(--+severity-code+---)--'|
```

If 00 were not a default value, it would be shown as follows:

```
>>+-----+-----<<
|'-PARMF(--+00-----+---)--'|
|'-severity-code-'|
```

(11) Qualified Object Names

Qualified object names include the optional library qualifier followed by the object name. If the qualifier is not specified, the default (*LIBL in this case) is used. Usually, *LIBL is the default value for a qualified object name; it means the system is to search the library list associated with the job to find the object.

For PARMD, the syntax diagram shows that a name can be specified in qualified or unqualified form. See (9) for coded examples.

(12) Positional Coding Limit and Key Parameter Limit

The point to which the command's parameters can be coded in positional form is indicated by a footnote number on the base line.

All parameters preceding the positional limit footnote can be coded without keywords (that is, in positional form); all parameters after the positional limit footnote must be coded with their keywords. If you attempt to code a parameter in positional form beyond this point, a syntax error will occur. When no parameters can be coded in positional form, a positional limit footnote either is not shown on the syntax diagram or it is shown directly following the command name.

A command can also have *key parameters*, which are the only significant parameters needed by the system to determine the actual values for other parameters on the command. Key parameters are the only parameters shown on the display when a user prompts for the command. After values are entered for the key parameters, the remaining parameters are shown with actual values instead of the default values (such as *SAME or *PRV).

Key parameters include all parameters preceding the key parameter limit, which is designated on the syntax diagram by a footnote number. If no key limit footnote is shown, or if it is shown directly following the command name, there are *no* key parameters on the command.

(13) Optional Parameters and Required Parameters

Optional parameters do not have to be coded. Optional parameters are shown below the base line as follows:

```
>>+-----+-----<<
|'-PARME(--+*BLANK-----+---)--'|
|'-message-text'-'|
```

If this parameter were required, it would appear on the base line as follows:

```
>>-PARME(---+*BLANK-----+---)-----><
         '- 'message-text'-'
```

In most cases, required parameters precede optional parameters on the syntax diagram.

(14) Quoted Values

User-defined variables that must be enclosed in apostrophes are shown enclosed with apostrophes in the diagram. Apostrophes are shown where one or more special characters are normally expected to occur among alphanumeric characters.

The value specified for PARME requires apostrophes if more than one word is entered or if special characters are used. Blanks, such as between words, are not allowed in an unquoted character string. PARMJ requires apostrophes if a character other than an alphanumeric character is specified.

```
PARME('This is a quoted string')
PARME('10-24-78')
PARME(102478)
```

The first and second values require apostrophes because they have either blanks (spaces) or special characters (-). The third value is a date with no separator characters; therefore, apostrophes are not required.

(15) Ordered List of Values

Ordered lists of values (that is, values that must be specified in a particular order) will be shown as follows:

```
>>-PARMG(---lower-value---upper-value---)-----><
```

Note: An ordered list of values can contain predefined values, see (17).

Both PARMG and PARMJ show a list of two values that must be coded in the order shown when the parameter is coded. Parentheses are required around the list of values even if no keyword is used.

PARMG and PARMJ could be coded as:

```
PARMG(0 16)          PARMJ(1 9999)
```

(16) Unordered List of Values (with repetition)

A parameter that can have several values specified in no particular order (an unordered list of like values) is shown as follows:

```

     .-*NONE-----
     |   .-----   |
     |   v           |   (1)
     |   |           |
>>-PARMH(-----+*VALUE1-----+-----)-----><
        +-*VALUE2----+
        +-*VALUE3----+
        '-user-value-'
```

Notes:

- 1. A maximum of 3 repetitions.

The PARML parameter can be specified as one, two, or three of the values shown. As the syntax note shows, any combination of three values selected and a user-defined value can be specified in any order. The user-defined value could be any value allowed for that parameter. Any of the following could be coded:

```
PARML(*VALUE1 *VALUE3)
PARML(*VALUE3 *VALUE2 16)
PARML(16 3 12)
```


Notice that in this example, that *NONE can be specified by itself, and the second element does not have to be specified. However, if the first element is coded by itself, the extra set of parentheses are required as shown below:

```
PARMQ(( *NO) (*YES *VAL3) (*YES))
```

(18) Ordered Lists of Values as Fragments

In this manual, fragments are used to show and reference long lists of values, as shown on (18) in the sample syntax diagram. If there are fewer than five elements, the structure shown for PARMN is used (see PARMN details on the sample syntax diagram). The structure of PARMN correctly shows that the elements must be coded in positional form. You cannot code *BLANK without coding the first three values or including the place holder, *N. The following could be entered:

```
PARMN(A)
PARMN(A B)
PARMN(A B GENVAL*)
PARMN(A *N GENVAL* *BLANK)
```

Note: You can use *N as a place holder.

However, the following would yield incorrect results because the elements are not coded in positional form.

```
PARMN(GENVAL* *BLANK)
PARMN(*BLANK)
PARMN(B 'User Object')
```

If a parameter contains five or more elements, the structure shown for PARMN is used (see PARMN details in the sample syntax diagram) because using the structure shown for PARMN for large numbers of elements makes the syntax diagram too complex. These elements are all shown as optional. However, they, like the elements in PARMN, must be coded in positional form.

Fragments can contain fragments as shown in PARMN details on the sample syntax diagram. PARMN details contains all of the PARMN details all of which must be specified in positional form.

(19) Dependent Parameter Relationships

Some parameters or values have dependent relationships with other parameters or values. Parameters that depend on values are indicated by a footnote. Parameters that are mutually exclusive are shown in the example below:

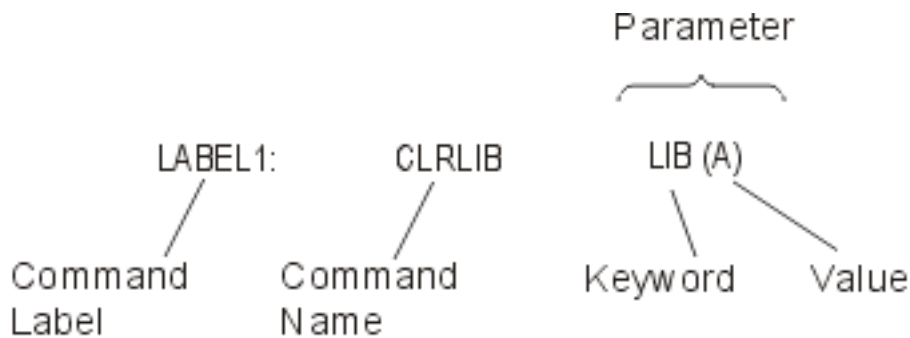
```
>>+-----+-----+-----+-----+-----+-----+-----+-----+-----+<<
|       .-*VALUE1--.              (1) (2)                               |
+-PARMI(--+-*VALUE2--+---)---PARMJ(-----value-----)---+
|       '-*VALUE3--'                                                    |
'-PARMK(--+-*NONE-----+---)-----+-----+-----+-----+-----+-----+
|                                       '-any-value-'                  |
```

Notes:

1. PARMJ is required only when PARMI(*VALUE2) is specified.
2. PARMJ and PARMK are mutually exclusive parameters; PARMJ and PARMK cannot be coded on the same command. As the note shows, PARMJ is a dependent parameter which is valid only when *VALUE2 is specified for the PARMJ parameter. Note that PARMJ cannot be specified if PARMK is specified. Also note that all of these parameters are optional because they appear below the base line.

Command parts

This figure illustrates the parts of a command, which includes: command label (optional), command name (mnemonic), and one or more parameters. The parameter includes a keyword and a value.



For information about the parts of a command, see the following:

- “Command label”
- “Command name”
- “Command parameters” on page 59

In addition, see the following for additional information about command coding:

- “Command syntax” on page 67
- “Command delimiters” on page 67
- “Command continuation” on page 69
- “Entering comments in commands” on page 69
- “Summary of general command coding rules” on page 70

For additional information about commonly used parameters, see “Commonly used parameters: Expanded descriptions” on page 144.

Command label

Command labels identify particular commands for branching purposes in a CL program. Labels can also be used to identify statements in CL programs that are being debugged. They can identify statements used either as breakpoints or as starting and ending statements for tracing purposes.

Typed just before the command name, the label can contain as many as 10 characters. The standard rules for specifying simple names (*SNAME) apply. The label, immediately followed by a colon and blanks (though not required), usually occurs between the colon and the command name. START: and TESTLOOP: are examples of command labels.

Command labels are not required, but a label can be placed on any command. Labels that are placed on commands that cannot be run (for example, the Declare CL Variable (DCL) command), the program branches to that label, the next command following the label is run. If the command following the label cannot be run, the program will move to the next command that can be run. Similarly, you can specify only one label on a line; if a command is not located on that line, the program will jump to the next command that can be run.

To specify multiple labels, each additional label must be on a separate line preceding the command as shown:

```

LABEL1:
LABEL2:  CMDX
  
```

No continuation character (+ or -) is allowed on the preceding label lines.

Command name

The command name identifies the function that will be performed by the program that is called when the command is run. The command name is an abbreviation of the command description. For example, the

name MOV OBJ identifies the CL command (Move Object) that moves an object from one library to another. Like other objects, a command name can be optionally qualified by a library name. See “Simple and qualified object names” on page 139 for more information about object names.

For more information about how CL commands are named, see “Object naming rules” on page 141.

Command parameters

Most CL commands have one or more *parameters* that specify the objects and values used to run the commands. When a command is entered, the user supplies the command object name, the parameter keyword names, and the parameter values used by the command. The number of parameters specified depends upon the command. Some commands (like the DO (Do) command and the ENDBCHJOB (End Batch Job) command) have no parameters, and others have one or more.

The specification of a group of values on one parameter is described under “List of values” on page 66.

In this topic, the word *parameter* usually refers to the combination of the parameter keyword and its value. For example, the Move Object (MOV OBJ) command has a parameter called OBJ that requires an object name to be specified. OBJ is the parameter keyword, and the name of the object is the value entered for the OBJ parameter.

See the following topics:

- “Required, optional, and key parameters”
- “Parameters in keyword and positional form”
- “Parameter values” on page 60

Required, optional, and key parameters: A command can have parameters that must be coded (required parameters) and parameters that do not have to be coded (optional parameters). Optional parameters are usually assigned a system-defined default value if another value is not specified for the parameter when the command is entered.

A command can also have *key parameters* which are the only parameters shown on the display when a user prompts for the command. After values are entered for the key parameters, the remaining parameters are shown with actual values instead of the default values (such as *SAME or *PRV).

Note: Key parameters include all parameters preceding the key parameter limit, which is designated on the syntax diagram by the letter **K**.

Parameters in keyword and positional form: You can specify parameters in CL using keyword (on page 59) form, positional (on page 59) form, or in a combination of the two.

Parameters in keyword form

A parameter in *keyword form* consists of a keyword followed immediately by a value (or a list of values separated by blank spaces) enclosed in parentheses. You cannot use blanks between the keyword and the left parenthesis preceding the parameter value. You can place blanks between the parentheses and the parameter value. For example, LIB(MYLIB) is a keyword parameter specifying that MYLIB is the name of the library that is used in some way, depending upon the command in which this LIB parameter is used.

When command parameters are all specified in keyword form, they can be placed in any order. For example, the following two commands are the same:

```
CRTLIB LIB(MYLIB) TYPE(*TEST)
CRTLIB TYPE(*TEST) LIB(MYLIB)
```

Parameters in positional form

A parameter in *positional form* does not have its keyword coded; it contains only the value (or values, if it is a list) whose function is determined by its position in the parameter set for that command. The parameter values are separated from each other and from the command name by one or more blank spaces. Because there is only one positional sequence in which parameters can be coded, the positional form of the previous CRTLIB (Create Library) command first example is:

```
CRTLIB MYLIB *TEST
```

If you do not want to enter a value for one of the parameters, the predefined value *N (null) can be entered in that parameter's position. The system recognizes *N as an omitted parameter, and either assigns a default value or leaves it null. In the previous CRTLIB command second example, if you coded *N instead of *TEST for the TYPE parameter, the default value *PROD is used when the command is run, and a production library named MYLIB is created with no public authority. The description of the CRTLIB command contains the explanation for each parameter.

Note: Parameters cannot be coded in positional form beyond the positional coding limit. If you attempt to code parameters in positional form beyond that point, the system returns an error message.

Combining keyword and positional forms

A command may also have its parameters coded in a combination of keyword and positional forms. The following examples show three ways to code the Declare CL Variable (DCL) command.

Keyword form:

```
DCL VAR(&QTY) TYPE(*DEC) LEN(5) VALUE(0)
```

Positional form:

```
DCL &QTY *DEC 5 0
```

Positional and keyword forms together:

```
DCL &QTY *DEC VALUE(0)
```

In the last example, because the optional LEN parameter was not coded, the VALUE parameter *must* be coded in keyword form.

Note: You cannot specify parameters positionally after a parameter specified in keyword form.

For additional information, see “Summary of general command coding rules” on page 70.

Parameter values: A parameter value is user-supplied information used during the running of a command. An individual value can be specified in any one of the following:

- “Constant value” on page 61 (its actual value): The types of constants are character string (includes names, date and hexadecimal values), decimal, and logical.
- “Variables” on page 65 name (the name of the variable containing the value): The types of variables are character string (includes names), decimal, and logical. Decimal and logical values must match the type of value expected for the parameter. Character variables can specify any type of value. For example, if a decimal value is expected, it can be specified by a character variable as well as by a decimal variable.
- “Expressions” on page 66 (the value used is the result of evaluating an expression): The types of expressions are arithmetic, character string, relational, and logical. Expressions can be used as a value for parameters in commands in CL programs only.
- “List of values” on page 66 (a list of values is one or more values that can be specified for a parameter): Not all parameters can accept a list of values. A *list parameter* can be defined to accept a specific set of multiple values that can be of one or more types. Values in the list must be separated by one or more blanks. Each list of values is enclosed by parentheses, indicating that the list is treated as a single

parameter. Parentheses are used even when a parameter is specified in positional form. To determine whether a list can be specified for a parameter, and what kind of list it can be, refer to the parameter description under the appropriate command description.

A parameter can specify one or a group of such values, depending on the parameter's definition in a command. If a group of values is specified, the parameter is called a *list parameter* because it can contain a list of values.

On commands with key and positional parameters, values can be specified in keyword form, positional form, or a combination of both forms. Parameter values must be enclosed in parentheses if any of the following conditions are true:

- A keyword precedes the value.
- The value is an expression.
- A list of values is specified.

Note: If only one value is specified for a list, no parentheses are required.

For additional information, see "Summary of general command coding rules" on page 70.

Constant value: A constant value is an actual numeric value or a specific character string whose value does not change. Three types of constants can be used by the control language: character string (quoted or unquoted), decimal, and logical.

For more information about constant value, see the following:

- "Character string" on page 61
- "Decimal values" on page 64
- "Logical values" on page 64

Character string

A *character string* is a string of any EBCDIC characters (alphanumeric and special) that are used as a value, including date and hexadecimal values. A character string can have two forms: quoted string or unquoted string. Either form of character string can contain as many as 5000 characters.

A *quoted* character string is a string of alphanumeric and special characters that are enclosed in apostrophes. For example, 'Credit limit has been exceeded' is a quoted character string.

The quoted string is used for character data that is not valid in an unquoted character string. For example, user-specified text can be entered in several commands to describe the functions of the commands. Those descriptions must be enclosed in apostrophes if they contain more than one word because blanks are not allowed in an unquoted string.

An *unquoted* character string is a string consisting of only alphanumeric characters and the special characters that are shown in the *Unquoted String* column in the Table 5 (on page 62). The table summarizes the main EBCDIC characters that are valid in unquoted and quoted character string values. An X in the last column indicates that the character on the left is valid; refer to the specific notes following the figure that indicate why the character is valid as described. The special characters allow the following to be unquoted character string values:

- Predefined values (* at the beginning)
- Qualified object names (/)
- Generic names (* at the end)
- Decimal constants (+, -, ., and ,)

Any of these unquoted strings can be specified for parameters defined to accept character strings. In addition, some parameters are defined to accept predefined values, names, or decimal values either singly or in combinations.

Table 5. Quoted and Unquoted Character Strings

Name of Character	Character	Unquoted String	Quoted String
Ampersand	&	See Note 5	X
Apostrophe	'	See Note 7	-
Asterisk (*)	*	See Notes 5, 6	X
At sign	@	X	X
Blank			X
Colon	:		X
Comma	,	See Note 1	X
Digits	0-9	See Note 1	X
Dollar sign	\$	X	X
Equal	=	See Notes 5, 8	X
Greater than	>	See Notes 5, 8	X
Left parenthesis	(See Note 4	X
Less than	<	See Notes 5, 8	X
Letters (lowercase)	a-z	See Note 2	X
Letters (uppercase)	A-Z	X	X
Minus	-	See Notes 1, 5	X
Not	¬	See Notes 5, 8	X
Number sign	#	X	X
Percent	%		X
Period	.	See Notes 1, 11	X
Plus	+	See Notes 1, 5	X
Question mark	?		X
Quotation marks	" "	See Note 10	X
Right parenthesis)	See Note 4	X
Semicolon	;		X
Slash	/	See Notes 3, 5	X
Underscore	_	See Note 9	X
Vertical bar		See Notes 5, 8	X

Notes:

1. An unquoted string of all numeric characters, an optional single decimal point (. or), and an optional leading sign (+ or -) are valid unquoted strings. Depending on the parameter attributes in the command definition, this unquoted string is treated as either a numeric or character value. On the CALL command or in an expression, this unquoted string is treated as a numeric value; a quoted string is required if character representation is desired. Numeric characters used in any combination with alphanumeric characters are also valid in an unquoted string.
2. In an unquoted string, lowercase letters are translated into uppercase letters unless the string is specified for a parameter that has the attribute CASE(*MIXED).
3. A slash can be used as a connector in qualified names and path names.
4. In an unquoted string, parentheses are valid when used to delimit keyword values and lists, or in expressions to indicate the order of evaluation.
5. In an unquoted string, the characters +, -, *, /, &, |, ¬, <, >, and = are valid by themselves. If they are specified on a parameter that is defined in the command definition with the EXPR(*NO) attribute, they are treated as character values. If they are specified on a parameter that is defined in the command definition with the EXPR(*YES) attribute, they are treated as expression operators.

6. In an unquoted string, the asterisk is valid when followed immediately by a name (such as in a predefined value) and when preceded immediately by a name (such as in a generic name). For further information on unquoted strings in expressions, see “Expressions” on page 66.
7. Because an apostrophe within a quoted string is paired with the opening apostrophe (delimiter) and is interpreted as the ending delimiter, an adjacent pair of apostrophes (‘’) must be used inside a quoted string to represent an apostrophe that is not a delimiter. When characters are counted in a quoted string, a pair of adjacent apostrophes is counted as a single character.
8. In an unquoted string, the characters <, >, =, ~, and | are valid in some combinations with another character in the same set. Valid combinations are: <=, >=, ~=, ~>, ~<, ||, |<, and |>. If the combination is specified on a parameter that is defined in the command definition with the EXPR(*NO) attribute, it is treated as a character value. If it is specified on a parameter that is defined in the command definition with the EXPR(*YES) attribute, it is treated as an expression operator.
9. In an unquoted string, the underscore is not valid as the first character or when used by itself.
10. Quotation marks are used to delimit a quoted name.
11. A period is valid in a basic name, except as the first character.

The following are examples of quoted string constants:

Constant	Value
'1,2,'	1,2,
'DON'T'	DON'T
'24 12 20'	24 12 20

The following are examples of unquoted strings:

Constant	Meaning
CHICAGO	CHICAGO
FILE1	FILE1
*LIBL	Library list
LIBX/PGMA	Program PGMA in library LIBX
1.2	1.2

More information and examples can be found in “Character string expressions” on page 79.

Date values

A date value is a character string that represents a date. Its format is specified by the system value QDATFMT. The length of the date value varies with the format used and whether a separator character is used. For example, if no separator character is used, the length of a date in a Julian format is five characters, and the length of a date in a non-Julian format is six characters. If a separator character is used, the length will be greater. More information on system value QDATFMT is in the Work Management



book.

The system value QDATSEP specifies the optional separator character that can be used when the date is entered. If a separator character is used, the date must be enclosed in apostrophes. For additional

information on system value QDATSEP, see the Work Management  book.

A date value can be specified for the parameters of type *DATE. A year value equal to or greater than 40 indicates a year from 1940 through 1999. A year value less than 40 indicates a year from 2000 through 2039. For additional information on parameter value *DATE, see the PARM statement description.

Hexadecimal values

A hexadecimal value is a constant made up of a combination of the hexadecimal digits A through F and 0 through 9. All character strings except names, dates, and times can be specified in hexadecimal form. To specify a hexadecimal value, the digits must be specified in multiples of two, be enclosed in apostrophes, and be preceded by an X. Examples are: X'F6' and X'A3FE'.

Note: Care should be used when entering hexadecimal values in the range of 00 through 3F, or the value FF. If these characters are shown or printed, they may be treated as device control characters producing unpredictable results.

Decimal values

A decimal value is a numeric string of one or more digits, optionally preceded by a plus (+) or minus (-) sign. A decimal value can contain a maximum of 15 digits, of which no more than nine can follow the decimal point (which can be either a comma or a period). Therefore, a decimal value can have no more than 17 character positions including the plus or minus sign and decimal point (if any). The following are examples of decimal values.

123.	} Equivalent Values	+ .017
1.23		6278,954374
1,23		-123456.987654321
-1,23		87654321.123

Logical values

A logical value is a single character (1 or 0) enclosed in apostrophes. It is often used as a switch to represent a condition such as on or off, yes or no, and true or false. When used in expressions, it can be optionally preceded by *NOT or ~. The following are examples of logical values:

Constant	Value	Meaning
'0'	0	Off, no, or false
'1'	1	On, yes, or true

Floating-point constants

A floating-point constant is a representation of a number that consists of:

- A significant sign: The significant sign may be + or -. The significant sign is optional; it is assumed to be + if no sign is specified.
- A significant: The significant must contain a decimal point. The maximum number of digits that can be specified for the significant is 253; however, only the first 17 significant digits are used.
- An exponent character: The exponent character must be E.
- An exponent sign: The exponent sign must be + or -. The significant sign is optional; it is assumed to be + if no sign is specified.
- An exponent: The exponent must be an integer; numbers 0 through 9 are valid. The maximum number of digits that can be specified is three.

All floating-point constants are stored as double-precision values. No blanks are allowed between any of the parts of a floating-point constant, and the parts must be in the order listed above.

Three commands have parameters for which floating-point constants can be specified:

- Call Program (CALL) command: You can use the PARM parameter to pass a floating-point constant to a called program. Any program you call must receive a floating-point constant as a double precision value.
- Change Program Variable (CHGPGMVAR) command: You can use the VALUE parameter to change a floating-point variable in a program.
- Copy File (CPYF) command: You can use floating-point construction in the FROMKEY, TOKEY, and INCREL parameters to select which records are copied from a database file.

For more information about floating-point constants see the DDS Reference information in the Database and Filesystems topic from the left navigation bar.

Variables: A *variable* contains a data value that can be changed when a program is run. The variable is used in a command to pass the value that it contains at the time the command is run. The change in value can result if one of the following conditions occur: the value is received from a data area, a display device file field, or a message; the value is passed as a parameter; a Change Variable (CHGVAR) command is run in the program; or another program that is called changes the value before returning it.

The variable name identifies a value to be used; the name points to where the actual data value is. Because CL variables are valid only in CL programs, they are often called *CL program variables* or, simply, CL variables. CL variable names must begin with an ampersand (&).

CL variables can be used to specify values for almost all parameters of CL commands. When a CL variable is specified as a parameter value and the command containing it is run, the current value of the variable is used as the parameter value. That is, the variable value is passed as if the user had specified the value as a constant.

Because it is generally true that CL variables can be used for most parameters of commands in CL programs, the command descriptions usually do not mention CL variables. For parameters that are restricted to constants only (such as in the DCL command), to CL variables only (such as all of the parameters of the Retrieve Job Attributes (RTVJOBA) command), or to specific types of variables (such as on the RTVJOBA or Retrieve Message (RTVMSG) command), the individual parameter descriptions specify those limitations. Otherwise, if the command is allowed in a CL program, CL variables can be used in place of a value, even with parameters that accept only predefined values. For example, a SAVE parameter having only predefined values of *YES and *NO can have a CL variable specified instead; its value can then be changed to *YES or *NO, depending on its value when the command is run.

A CL variable must contain only one value; it may not contain a list of values separated by blanks.

The value of any CL program variable can be defined as one of the following types:

- Character: A character string that can contain a maximum of 9999 characters. The character string can be coded in quoted or unquoted form, but only the characters in the string itself are stored in the variable.
- Decimal: A packed decimal value that can contain a maximum of 15 digits, of which no more than nine can be decimal positions.
- Logical: A logical value of '1' or '0' that represents on/off, true/false, or yes/no.

If value is:	CL variable can be declared as:
Name	Character
Date or time	Character
Character string	Character
Numeric	Decimal or character
Logical	Logical or character

Expressions: An expression is a group of constants or variables, separated by operators, that produces a single value. The operators specify how the values are combined to produce the single value or result. The operators can be arithmetic, character string, relational, or logical. The constants or variables can be character, decimal, or logical. For example, the expression (&A + 1) specifies that the result of adding 1 to the value in the variable &A is used in place of the expression.

Character string expressions can be used in certain command parameters defined with `EXPR(*YES)` in CL programs. An expression can contain the built-in functions `%BINARY` (or `%BIN`), `%SUBSTRING` (or `%SST`), and `%SWITCH`, which are covered in detail in “Expressions in CL commands” on page 78. The types of expressions and examples of each are described there.

List of values: A list of values is one or more values that can be specified for a parameter. Not all parameters can accept a list of values. A *list parameter* can be defined to accept a specific set of multiple values that can be of one or more types. Values in the list must be separated by one or more blanks. Each list of values is enclosed by parentheses, indicating that the list is treated as a single parameter. Parentheses are used even when a parameter is specified in positional form. To determine whether a list can be specified for a parameter, and what kind of list it can be, refer to the parameter description under the appropriate command description.

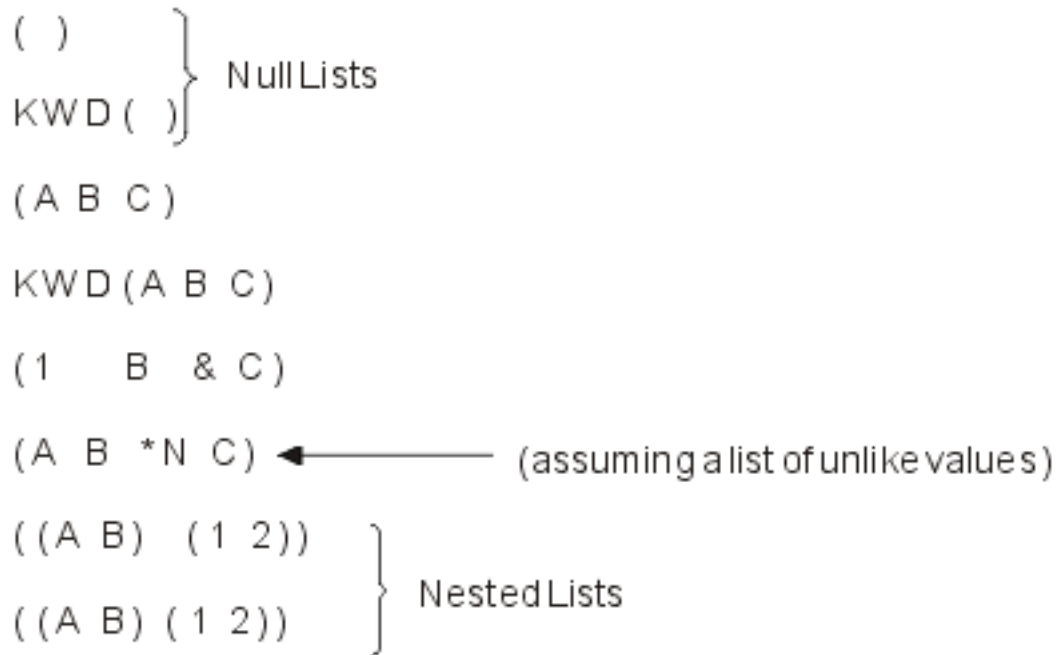
A list parameter can be defined to accept a list of multiple like values (a simple list) or a list of multiple unlike values (a mixed list). Each value in either kind of list is called a *list element*. List elements can be constants, variables, or other lists; expressions are not allowed.

- A *simple list* parameter accepts one or more values of the type allowed by a parameter. For example, (RSMITH BJONES TBROWN) is a simple list of three user names.
- A *mixed list* parameter accepts a fixed set of separately defined values that are in a specific order. Each value can be defined with specific characteristics such as type and range. For example, `LEN(5 2)` is a mixed list in which the first element (5) gives the length of a field and the second element (2) gives the number of decimal positions in that field.
- For many parameters defined to accept lists, predefined single values can be specified in place of a list of values. One of these single values can be the default value, which can be either specified or assumed if no list is specified for a simple or mixed list. To determine what defaults are accepted for a given list parameter, refer to the description of the parameter in the description of the command for which the parameter is defined and used.

Note: *N cannot be specified in a simple list, but it can be specified in a mixed list. Also, individual parameters passed on the `CALL` command cannot be lists.

- The maximum level of nesting of lists inside lists is three, including the first. These are indicated by three nested levels of parentheses.

The following are examples of lists:



The last two examples contain two lists nested inside a list: the first list contains values of A and B, and the second list contains values of 1 and 2. The space between the two nested lists is not required. Blanks are the separators between the values inside each nested list, and the sets of parentheses group the nested values into larger lists.

Command syntax: Commands have the following general syntax. The brackets indicate that the item within them is optional; however, the parameter set may or may not be optional, depending upon the requirements of the command.

```
[//] [?] [label-name:] [library-name/] command-name
[parameter-set]
```

Note: The // is valid only for a few batch job control commands, such as the DATA command. The // identifies those types of commands sent to the spooling reader that reads the batch job input stream.


Command delimiters: Command delimiters are special characters or spaces that identify the beginning or end of a group of characters in a command. Delimiters are used to separate a character string into the individual parts that form a command: command label, command name, parameter keywords, and parameter values. Parameter values can be constants, variable names, lists, or expressions. The following diagram shows various delimiters for a command:



The following delimiters are used in the OS/400 control language:

- The colon (:) ends the command label, and it can be used to separate the command label from the command name.
- Blank spaces separate the command name from its parameters and separate parameters from each other. They also separate values in a list. Multiple blanks are treated as a single blank except in a quoted character string or comment enclosed in apostrophes. A blank *cannot* separate a keyword and the left parenthesis for the value.
- Parentheses () are used to separate parameter values from their keywords, to group lists of values, and to group lists within lists.
- Slashes (/) connect the parts of a qualified name or the parts of a path name.
 - For a qualified object name, the two parts are the library qualifier and the object name (LIBX/OBJA). Qualified object names are described in Simple and qualified object names.
 - For a path name, the parts are the directory or directories searched and the object name ('/Dir1/Dir2/Dir3/ObjA').
- Either a period or a comma can be used as a decimal point in a decimal value (3.14 or 3,14). Only one decimal point is allowed in a value.
- Apostrophes specify the beginning and ending of a quoted character string, which is a combination of any of the 256 extended binary-coded decimal interchange code (EBCDIC) characters that can be used as a constant. For example, 'YOU CAN USE \$99@123.45 (*)></ and lowercase letters' is a valid quoted string that is a constant.

Because an apostrophe inside a quoted string is paired with the opening apostrophe (delimiter) and is interpreted as the ending delimiter, an apostrophe inside a quoted string must be specified as two apostrophes. A pair of adjacent apostrophes used this way is counted as a single character.

- A special character is used to separate a date into three parts: month, day, and year (two parts for Julian dates: year and day). The special characters that may be used as date separators are the slash (/), the hyphen (-), the period (.), and the comma (,). The special character used to code as separators in a command date must be the same as the special character specified as the date separator for the job. The Work Management  book contains more information on specifying job and system values.
- The characters /* and */ can indicate the beginning and ending of a comment, or can be used in a character string. To begin a comment, the characters /* must begin in the first position of the command, be preceded by a blank, or be followed by either a blank or an asterisk. If the characters /* or */ occur in a later position of a command, they will usually be enclosed in apostrophes and can represent, for example, all objects in the current directory for a path name. For more information on path names, see the Integrated File Systems topic.
- A question mark (?) preceding the command name indicates that the command is prompted. If the command is specified with a label, the question mark may either precede the label, or it may follow the label and precede the command name.

Within a CL program, when a question mark precedes a command name, a prompt display is presented. You can enter parameter values not specified on the command in the program.

Prompting characters may be put into a command in two forms. A single question mark (?) may be coded before the command name (either before or after the command label in a CL program) to cause the entire command to be prompted. Selective prompt characters (?? or ?*) may be coded before any parameter keyword to cause that parameter to be prompted when the command is run.

If a question mark is entered before the command name on the command entry display, the effect is the same as pressing the F4 (Prompt) key after the command is entered.

Within a CL program, when a question mark precedes the command name, a prompt display is presented. This display is of the same format as that presented when pressing the F4 key from the command entry display. Parameters of the command for which the program has coded values are shown for informational purposes, but the user cannot change the values supplied by the program. Parameters for which no value was coded are shown as input fields so you can enter values to be used in processing the command.

Selective prompting allows you to identify specific command parameters to be prompted. To call selective prompting, the characters ??, ?*, or ?- are coded immediately preceding the keyword name of the parameter(s) to be prompted. More information on prompting is available in the CL Programming



book.

Selective prompting is not allowed with command string (*CMDSTR) parameters.

Note: Parameters of the command that are preceded by the characters ?* are shown, but you cannot change the values that are supplied by the program. Parameters preceded by the characters ?? are shown as input fields containing the values coded in the program or command defaults so you can enter or change the values used in processing the command. Parameters preceded by the characters ?- are omitted from the display. All selectively prompted parameters must be coded in keyword or keyword-with-value form. Several parameters may be selectively prompted within one command. When selective prompting is called, only keywords that are immediately preceded by the selective prompt characters are prompted. All other parameters are processed using the values as coded on the command or, if not coded, using command defaults.

Either form of prompting, but not both, is allowed on a single command in a CL program. If the character ? precedes the command name and selective prompt characters (except ?-) precede any keyword, an error message is returned and the program is not created.

For additional information, see “Summary of general command coding rules” on page 70.

Command continuation: Commands can be entered in free format. This means that a command does not have to begin in a specific location on a coding sheet or on the display. A command can be contained entirely in one record, or it can be continued on several lines or in several records. Whether continued or not, the total command length cannot exceed 20,000 characters. Either of two special characters, the plus sign (+) or the minus sign (-), is entered as the last nonblank character on the line to indicate that a command is continued. Blanks immediately *preceding* a + or - sign are always included; blanks immediately following a + or - in the *same record* are ignored. Blanks in the *next record* that precede the first nonblank character in the record are ignored when + is specified but are included when - is specified.

The + is generally useful between parameters or values. At least one blank must precede the sign when it is used between separate parameters or values. The difference between the plus and minus sign usage is particularly important when continuation occurs inside a quoted character string. The following example shows the difference:

```
CRTLIB LIB(XYZ) TEXT('This is CONT+
  INUED')
```

```
CRTLIB LIB(XYZ) TEXT('This is CONT-
  INUED')
```

```
For + : CRTLIB LIB(XYZ) TEXT('This is CONTINUED')
```

```
For - : CRTLIB LIB(XYZ) TEXT('This is CONT  INUED')
```

Notes:

- The minus sign causes the leading blanks on the next line to be entered.
- Use continuation characters + and - in CL programs only. An error occurs if + or - is used on a command entry display.
- The characters + and - are used for multiple-command examples, but not for single-command examples.

Entering comments in commands: Comments can be inserted either inside or outside a command's character string wherever a blank is permitted. However, because a continuation character must be the last nonblank character of a line (or record), comments *may not* follow a continuation character on the same line.

For readability, it is recommended that each comment be specified on a separate line preceding or following the command it describes, as shown here:

```
MOV OBJ  OBJA  TOLIB(LIBY)
      /* Object OBJA is moved to library LIBY. */
DLT LIB  LIBX
      /* Library LIBX is deleted. */
```

Comments can include any of the 256 EBCDIC characters. However, the character combination `*/` should not appear within a comment because these characters end the comment. To begin a comment, the characters `/*` must be placed in the first position of the command, be preceded by a blank, or be followed by either a blank or an asterisk.

Summary of general command coding rules: This section contains a summary of general information needed to properly code CL commands.

Delimiters

- Blanks are the basic separators between the parts of a command:
 - Between command label and command name (not required, because the colon `[:]` is the delimiter).
 - Between command name and first parameter, and between parameters.
 - Between values in a list of values (not required between ending and beginning parentheses that enclose nested lists inside a list).
 - Between the slashes and the name or label of some job control commands, like `// DATA` (not required).
- Blanks *cannot* separate a parameter's keyword from the left parenthesis preceding its values. When a keyword is used, parentheses must be used to enclose the values; blanks *can* occur between the parentheses and the values. For example, `KWD(A)` is valid.
- Multiple blanks are treated as a single blank, unless they occur in a quoted string or a comment.
- A colon must immediately follow a command label. Only one label can be used on any command (`LABEL1: DCLF`).
- Apostrophes must be used to specify the beginning and end of a quoted character string. If a character string contains special characters, such as blanks, apostrophes are required. If an apostrophe must be used in the quoted string, two apostrophes must be entered consecutively to indicate that it is an apostrophe and not the end of the quoted string.
- Parentheses must be used:
 - On parameters that are specified (coded) in keyword form
 - To group multiple values in a single list, in a positional parameter, or around expressions
 - To indicate a list (of none, one, or several elements) nested inside *another* list
- Sets of parentheses inside parentheses can be entered as long as they are paired, up to the maximum of five nested levels in logical expressions or three nested levels in lists of values.
- Comments can appear wherever blanks are permitted, except after a continuation character on the same line or record.
- A plus or minus sign at the end of a line indicates that the command is continued on the following line. Blanks following a `+` or `-` sign in the same record are ignored; blanks in the next record that precede the first nonblank character are ignored when `+` is specified and included when `-` is specified. One blank must precede the `+` sign when it is used between separate parameters or values.

Parameters

- All required parameters must be coded.
- If an optional parameter is not coded, the system uses its default value if the parameter has one. In the syntax diagram of each command, default values are indicated by showing the default value above the line the keyword name is on. If no default value is indicated, the default varies (depending on other parameter values) and is described in the text, or the action taken does not require that parameter.

- Words or abbreviations specified in capital letters in the command and parameter descriptions must be coded as shown. This is true of all command names, the keywords of parameters (if used), and many parameter values. Lowercase letters not coded in quoted strings or in comments are translated to uppercase letters. Lowercase letters specified for values on parameters defined as CASE(*MIXED) are not translated to uppercase letters.
- If there are key parameters, the values for the key parameters must be entered on the prompt before the remaining parameters will be shown. Parameters preceding the key parameter footnote on the syntax diagram are key parameters.
- Parameters cannot be coded in positional form past the positional parameter footnote found in the syntax diagrams (if applicable). The order of positional coding is the order in which the parameters are presented in the syntax diagram and in the command description.

Parameter Values

- The first character in all names must be an alphabetic character (A through Z, \$, #, @, or a double quotation mark (")). Names must not exceed 10 characters (CL variable names and built-in function names can have 11 characters maximum, including the preceding & or % characters). In some commands, the names of objects can be specified in qualified form (library-name/object-name) or in path name form (directory-name/object name).
- Predefined values that begin with an asterisk can be used only for the purposes intended, unless they are included in comments or quoted strings. These include predefined parameter values (*ALL, for example), symbolic operators (*EQ, for example), and the null value (*N).
- In a CL program, a variable can be specified for all parameters, except where that is explicitly restricted. The user-specified value of the variable is passed as if it had been specified on the command.
- In a CL program, a character string expression can be specified for any parameter defined with EXPR(*YES). The resulting value of the expression is passed as if the value had been specified on the command.
- Null (omitted) values are specified with the characters *N, which mean that no value was specified and the default value, if one exists, should be used. *N is needed only when another value following the omitted value is being specified as a positional parameter or an element in a list.
- Either a comma or a period can be used to indicate a decimal point in a numeric value. The decimal point is the only special character allowed between digits in the numeric string; there is no delimiter for indicating thousands.
- When repetition is indicated for a parameter:
 - A predefined value is not coded more than once in a series of values.
 - As many user-defined values (like names or numeric limits) can be entered as there are different values or names, up to the maximum number of repetitions allowed. For example, if a syntax footnote states "A maximum of 20 repetitions," you can specify a maximum of 20 values.

Note: When you are using parameters that have the same name in different commands, the meaning of (and the values for) that parameter in each command may be somewhat different. Refer to the correct command description for the explanation of the parameter you are using. For some parameters, you can also refer to "Commonly used parameters: Expanded descriptions" on page 144 for both general information about a parameter and an expanded description of its values coded in commands.

CL programs

A Control Language (CL) program is a program that is created from source statements consisting entirely of control language commands.

See the following for concepts relating to CL programs:

- "CL character sets and values" on page 72
- "Naming within commands" on page 74
- "Folder and document names" on page 76

- “Expressions in CL commands” on page 78

CL character sets and values: This section explains the usage of the EBCDIC character sets, special characters, and IBM-defined fixed values called predefined values.

See the following:

- “Character sets”
- “Special character use”
- “Predefined values” on page 74

Character sets: The control language uses the extended binary-coded decimal interchange code (EBCDIC) character set. For convenience in describing the relationship between characters used in the control language and those in the EBCDIC character set, the following control language categories contain the EBCDIC characters shown:

Category	Characters Included
Alphabetic ¹	26 letters (A through Z), \$, #, and @
Numeric	10 digits (0-9)
Alphanumeric ^{1,2}	A through Z, 0 through 9, \$, #, @, period (.), and underscore (_)
Special Characters	All other EBCDIC characters (for those having special uses in CL, see “Special character use”)

Note:

- ¹ Lowercase letters (a through z) are accepted, but they are translated into the corresponding uppercase letters by the system except when they are included within a quoted character string or a comment, or they are specified for a value on a parameter that has the character (*CHAR) or the path name (*PNAME) attribute for its TYPE parameter and the mixed case (*MIXED) attribute for its CASE parameter in the command definition.
- ² The underscore (_) is an alphanumeric connector that can be used in iSeries 400 CL to connect words or alphanumeric characters to form a name (for example, PAYLIB__01). This use of the underscore might not be valid in other high-level languages.

The first three categories contain the characters that are allowed in quoted and unquoted character strings, in comments, and in CL names, such as in names of commands, labels, keywords, variables, and OS/400 objects. Special characters in the last category can only be used in quoted character strings and comments; they cannot be used in unquoted strings. However, some have special syntactical uses when coded in the proper place in CL commands. These uses are given in the “Special character use” chart.

Special character use: The following special EBCDIC characters are used by the CL in various ways. They are most frequently used as delimiters (which are covered in Delimiters under Command syntax) and as symbolic operators in expressions.

For more information about symbolic operators, see:

- “Symbolic operators” on page 73
- the discussion about the types of expressions in the “Operators in expressions” on page 82

Special characters can only be used in these special ways or inside quoted character strings or comments. The special characters, as shown in the following table, have assigned meanings when coded in control language commands:

Delimiters

Name	Symbol	Meanings
Apostrophe	' '	Single apostrophe delimiters indicate the beginning and end of a quoted character string (a constant).

Name	Symbol	Meanings
Begin and end comment	/* */	Indicates the beginning and end of a comment.
Blank	␣ ¹	Basic delimiter for separating parts of a command (label, command name, and its parameters), and for separating values inside lists.
Colon	:	Ending delimiter for command labels. Separates parts of time values. ³
Comma	,	In many countries, used as decimal point in numeric values. Separates parts of date values. ²
Left and right parentheses	()	Grouping delimiter for lists and parameter values, and for evaluating the order of expressions.
Period	.	Decimal point. Used to separate the name and extension of a document and folder name and to separate the parts of date values. ²
Quote	" "	Start of a quoted object name.
Slash	/	Connects parts of qualified names or path names.
Slashes	//	Identifying characters used in positions 1 and 2 of BCHJOB, ENDBCHJOB, and DATA commands in the job stream. Also, used as a default delimiter on inline data files.

Notes:

- ¹ Because this character does not resolve in the online version of this book, ␣ is used to represent a blank space only when the character cannot be clearly explained in another manner.
- ² Valid only when the job date separator value specifies the same character.
- ³ Valid only when the job time separator value specifies the same character.

Symbolic operators

The following characters are used as symbolic operators in CL commands.

Name	Symbol	Meanings
And	&	Symbolic logical operator for AND.
Asterisk	*	Multiplication operator. Indicates a generic name when it is the last character in the name. Indicates OS/400 reserved values (predefined parameter values and expression operators) when it is the first character in a string.
Concatenation	>, <, and ³	Character string operator (indicates both values are to be joined). See "Expressions" on page 66 for more information on the differences between these concatenation operators.
Equal	=	Symbolic <i>equal</i> relational operator.
Greater than	>	Symbolic <i>greater than</i> relational operator.
Less than	<	Symbolic <i>less than</i> relational operator.
Minus (hyphen)	-	Subtraction operator, command continuation operator, and negative signed value indicator. Separates parts of date values. ¹
Not	¬ ²	Symbolic NOT relational operator.
Or	³	Symbolic logical operator for OR.
Plus	+	Addition operator, command continuation character, and positive signed value indicator.
Slash	/	Division operator. Separates parts of date values. ¹ Used as the separator between parts of a qualified name.

Note:

- ¹ Valid only when the job date separator value specifies the same character.
- ² In some character sets, including the multinational character set, the character ^ replaces the ¬ character. Either ^ or *NOT can be used as the logical NOT operator in those character sets.
- ³ In some character sets, including the multinational character set, the character ! replaces the | character. Either ! or *OR can be used as the logical OR operator, and either || or *CAT can be used as the concatenation operator in those character sets.

Note: The symbolic operators can also be used in combinations as listed in the chart under Operators in expressions in Expressions.

Symbolic operators: Other uses

Symbolic operators can also be used in the following ways:

Name	Symbol	Meanings
Ampersand	&	Identifies a CL variable name when it is the first character in the string.
Percent	%	Identifies a built-in system function when it is the first character in the string.
Question mark	?	Specifies a prompt request when it precedes a command name or keyword name.

Predefined values: Predefined values are IBM-defined fixed values that have predefined uses in the CL and are considered to be reserved in the OS/400 system. Predefined values have an asterisk (*) as the first character in the value followed by a word or abbreviation, such as *ALL or *PGM. The purpose of the * in predefined values is to prevent possible conflicts with user-specified values, such as object names. Each predefined value has a specific use in one or more command parameters, and each is described in detail in the command description.

Some predefined values are used as operators in expressions, such as *EQ and *AND. The predefined value *N is used to specify a null value and can be used to represent any optional parameter. A *null value* (*N) indicates a parameter position for which no value is being specified; it allows other parameters that follow it to be entered in positional form. To specify the characters *N as a character value (not as a null), the string must be enclosed in apostrophes ('*N') to be passed. Also, when the value *N appears in a CL program variable at the time it is run, it is always treated as a null value.

Naming within commands: The type of name you specify in the OS/400 control language determines the characters you can use to specify a name. For certain types of names, there are restrictions on the use of certain characters to represent the name. Those types of names are *NAME, *SNAME, and *CNAME.

The *PNAME and *GENERIC list element or parameter types are only discussed briefly in this section.

Note: For a description of how to specify these names when you use command definitions to create commands, see the PARM (parameter) and ELEM (element) statements in “Command definition statements” on page 87.

The characters allowed for the *NAME, *SNAME, and *CNAME names and the rules you use to specify them are shown in the table and descriptions that follow.

Table 10. Allowable Characters for *NAME, *SNAME, and *CNAME

Type of Name	First Character	Other Characters	Min. Length	Max. Length
*NAME ¹	A-Z, \$, #, @	A-Z, 0-9, \$, #, @, _, .	1	256

Type of Name	First Character	Other Characters	Min. Length	Max. Length
*SNAME ¹	A-Z, \$, #, @	A-Z, 0-9, \$, #, @, _	1	256
*CNAME ¹	A-Z, \$, #, @	A-Z, 0-9, \$, #, @		
Quoted name ²	“ ³	Any except blank, *, ?, ', ", X'00'-X'3F', and X'FF'	3	256

Notes:

- ¹ The system converts lowercase letters to uppercase letters.
- ² Double quotes can only be used for basic names (*NAME).
- ³ Both the first and last characters must be a double quote (“).

***NAME (Basic Name):** Every basic name can begin with the characters A-Z, \$, #, or @ and can be followed by up to nine characters. The remaining characters can include the same characters as the first but can also include numbers 0-9, underscores (_), and periods (.). Lowercase letters are changed to uppercase letters by the system. Basic names used in IBM-supplied commands can be no longer than 10 characters. However, in your own commands, you can define parameters of type *NAME (specified on the TYPE parameter of the PARM or ELEM statements) with up to 256 characters.

Examples of basic names are shown below:

```
A987@.442#    ONE_NAME    LIBRARY_0690    $LIBX
```

Names can be entered in quoted or unquoted form. If you use the quoted form, the following rules and considerations also apply:

***NAME (Basic Name in Quoted Form):** Every quoted name must begin and end with a quotation mark ("). The middle characters of a quoted name can contain any character except , *, ?, ', ", hex 00 through 3F, or hex FF, and is delimited by a slash. Quoted names allow you to use graphic characters in the name. The quoted form of basic names used in IBM-supplied commands can be no longer than 8 characters (not including the double quotes). In your own commands, you can define parameters of type *NAME in quoted form with up to 254 characters (not including the double quotes).

Note: Only basic names can be used in quoted form.

Examples of quoted names are shown below:

```
"A"          "AA%abc"        "ABC%%abc"
```

When you use quoted names, you should be aware of certain restrictions:

- Code points in a name might not be addressable from all keyboards.
- Characters in a quoted name might not be valid in a high-level language.
- The System/38 environment supports only simple (*SNAME) names. If other characters are used, the objects cannot be accessed as System/38 environment objects.
- Names that are longer than eight characters cannot be accessed by the System/36 environment unless control language overrides are used.

- A Structured Query Language (SQL) name that contains a period must be specified in an SQL statement in quotation marks.

If a name enclosed in quotation marks is a valid unquoted basic name, the quotation marks are removed. Thus, "ABC" is equivalent to ABC. Because the quotation marks are removed, they are not included in the length of the name. "ABCDEFGHIJ" is, therefore, a valid name on IBM* commands even though it is longer than 10 characters.

***SNAME (Simple Name):** Simple names are the same as *unquoted* basic names but with one exception: periods (.) cannot be used. Simple names are used for CL variables, labels, and keywords to simplify the syntax of the control language.

Some examples of simple names are as follows:

```
NEWCMD    LIB_2
```

***CNAME (Communications Name):** Communications names are the same as *unquoted* basic names with the following exceptions:

1. Periods (.) and underscores (_) cannot be used.
2. For IBM commands, *CNAME is limited to 8 characters.

An example of a communications name is shown below:

```
APPN3@@
```

Note: Because restricted character sets are sometimes used by other IBM systems, use caution when choosing names that use the special characters #, \$, and @. These special characters might not be on the remote system's keyboard. The names that may be exchanged with the remote systems include the following:

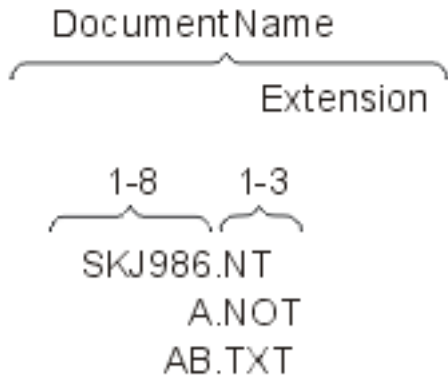
- Network IDs
- Location names
- Mode names
- Class-of-service names
- Control point names

Folder and document names: Folder names should describe the contents of a folder. The names must be unique and should be easy to type, as well as descriptive to a user. To find a particular folder on the system and change a document stored in it, you must either supply the folder name or select it from a list of names.

Document names should describe the contents of the document. You should give careful consideration to the names you use to help you find the document later. The names must be unique in the folder and should be easy to type, as well as descriptive.

The name you use for a folder or a document must follow these rules:

- The name must be unique within a folder.
- A document or folder name can be 1 to 12 characters long, including an optional extension. If no extension is included, a document or folder name can have a maximum of eight characters. If an extension is included, the extension must start with a period and can have up to three additional characters. An extension in the document name allows you to identify the document by using specific information that can help you do a selective listing of documents on your system.



- A document or folder name can include any single-byte EBCDIC character *except for the following* special characters that the system uses for other purposes:

Character

Asterisk (*)

Slash (/)

Question Mark (?)

Special uses

Multiplication operator, indicates generic names, and indicates OS/400 reserved values

Division operator, delimiter within system values, and separates parts of qualified object names

Initiates requests for system help

- When a folder is stored in another folder, both folder names are used, separated by a slash (/). That combination of names is called a **folder path**. For example, if a folder named FOLDR2 is stored in FOLDR1, the path for FOLDR2 is FOLDR1/FOLDR2. FOLDR1 is the **first-level folder**. FOLDR2 is the **next-level folder**. The name of a single folder can be 1 to 12 characters long, including an optional extension. A folder path can contain a maximum of 63 characters.


Folder names should not begin with Q because the system-supplied folder names begin with Q. The following are examples of permitted folder names and folder paths:

```
@LETTERS
FOLDER.PAY
PAYROLL/FOLDER.PAY
#TAX1/FOLD8.TAX/$1988/PAYROLL/FOLDER.PAY
```

Notes:

1. In CL commands, folder path names must be enclosed in apostrophes to prevent the system from processing them as qualified (library/object) names. If an apostrophe is to be part of the name, it must be specified as two consecutive apostrophes.
2. A number of CL commands act on either documents or folders, and some act on both. The abbreviation DLO (document library object) is used when referring to either a document or folder.
3. In CL commands, folder and document names must be enclosed in apostrophes if they contain characters that are CL delimiters.
4. The system does not recognize graphic characters; it recognizes only code points and uses the following assumptions:
 - All folder and document names are encoded using single-byte EBCDIC code pages. Since code points hex 41 through FE represent graphic characters in those code pages, they are the only code points that can be used in folder and document names.
 - Code points hex 5C, 61, and 6F represent the asterisk (*), slash (/), and question mark (?) respectively, and cannot be used in folder and document names.
 - The code points for lowercase letters in English (hex 81 through 89, 91 through 99, and A2 through A9) are converted to the code points for uppercase letters (C1 through C9, D1 through D9, and E2 through E9, respectively).

More information on code pages that are supported on the iSeries 400 is in Chapter 5 of the Local

Device Configuration  book and in the description of the CRTDEVDSP (Create Device Description (Display)) command.

In addition to the folder and document names previously described, folders and documents are internally classified in the system by their system object names. These are 10-character names derived from date/time stamps, and, while they are generally not known to the user, they may be specified on some CL commands by specifying *SYSOBJNAM for the folder or document name and by specifying the system object name in a separate parameter.

For more information, see “Naming within commands” on page 74.

Expressions in CL commands: A character string expression can be used for any parameter, element, or qualifier defined with EXPR(*YES) in the command definition object. Any expression can be used as a single parameter in the Change Variable (CHGVAR) and If (IF) commands. An expression in its simple form is a single constant, a variable, or a built-in function. An expression usually contains two operands and an operator that indicates how the expression is to be evaluated. Two or more expressions can be combined to make a complex expression.

The following types of expressions are supported in CL programs:

- “Arithmetic expressions” on page 78 (&VAR + 15)
- “Character string expressions” on page 79 (SIX || TEEN)
- “Relational expressions” on page 81 (&VAR > 15)
- “Logical expressions” on page 81 (&VAR & &TEST)

A *complex* expression contains multiple operands, operators that indicate what operation is performed on the operands, and parentheses to group them. Only one operator is allowed between operands, except for the + and - signs when they immediately precede a decimal value (as a signed value), and the *NOT operator when it is used in a logical expression.

No complex expression can have more than five nested levels of parentheses, including the outermost (required) level.

Arithmetic and character string expressions can be used together in a complex expression if they are used with relational and logical operators; for example: (A=B&(1+2)=3). A pair of arithmetic expressions or a pair of character string expressions can be compared within a relational expression. Also, relational expressions can be used within a logical expression

Arithmetic expressions

The operands in an arithmetic expression must be decimal constants or decimal CL variables. An arithmetic operator (only in symbolic form) must be between the operands. The results of all arithmetic expressions are decimal values, which may be stored in a CL variable.

Note: The division operator (/) must be preceded by a blank if the operand that precedes it is a variable name. (For example, &A /2, *not* &A/2.) All other arithmetic operators may optionally be preceded or followed by a blank.

Arithmetic operands can be signed or unsigned; that is, each operand (whether it is a numeric constant or a decimal CL variable) can be immediately preceded by a plus (+) or minus (-) sign, but a sign is not required. When used as a sign, no blanks can occur between the + or - and its value. For example, a decimal constant of 23.7 can be expressed as +23.7 or -23.7 (signed) or as 23.7 (unsigned).

The following are examples of arithmetic expressions:

(&A + 1)
(&A - &F)
(&A + (-&B))

(&A + &B -15)
(&A+&B-15)

If the last nonblank character on a line is a + or -, it is treated as a continuation character and not as an arithmetic operator.

Character string expressions

The operands in a character string expression must be quoted or unquoted character strings, character variables, or the substring (%SUBSTRING or %SST) built-in function. The value associated with each variable or built-in function must be a character string. The result of concatenation is a character string.

There are three operators that can be used in character string expressions. These operators concatenate (or join) two character strings, but each has a slightly different function. They are:

- *CAT (concatenation, symbol ||)

***CAT Operator**

The *CAT operator concatenates two character strings. For example:

ABC *CAT DEF becomes ABCDEF

Blanks are included in the concatenation. For example:

'ABC ' *CAT 'DEF ' becomes 'ABC DEF '

- *BCAT (concatenation with blank insertion, symbol |>)

***BCAT Operator**

The *BCAT operator truncates all trailing blanks in the first character string; one blank is inserted, then the two character strings are concatenated. Leading blanks on the second operand are not truncated.

For example:

ABC *BCAT DEF becomes ABC DEF

'ABC ' *BCAT DEF becomes 'ABC DEF'

- *TCAT (concatenation with trailing blank truncation, symbol |<)

***TCAT Operator**

The *TCAT operator truncates all trailing blanks in the first character string, then the two character strings are concatenated. All leading blanks on the second operand are not truncated. For example:

ABC *TCAT DEF becomes ABCDEF

'ABC ' *TCAT DEF becomes 'ABCDEF'

ABC *TCAT ' DEF' becomes 'ABC DEF'

'ABC '*TCAT ' DEF' becomes 'ABC DEF'

All blanks that surround the concatenation operator are ignored, but at least one blank must be on each side of the reserved value operator (*CAT, *BCAT, or *TCAT). If multiple blanks are wanted in the expression, a quoted character string (a character string enclosed within apostrophes) must be used.

See the following examples for more information about character string expressions.

- "Example: Character string expressions"
- "Example: Using character strings and variables" on page 80

Example: Character string expressions

The following are examples of string expressions. Assume the following variables:

Variable	Value
&AA	'GOOD '
&BB	'REPLACEMENT'

Variable	Value
&CC	'ALSO GOOD'
&DD	'METHOD'

Expression	Result
(&AA &BB)	GOOD REPLACEMENT
(&AA &BB)	GOOD REPLACEMENT
(&AA *CAT &BB)	GOOD REPLACEMENT
(&CC > &DD)	ALSO GOOD METHOD
(&CC *BCAT &DD)	ALSO GOOD METHOD
(A *CAT MOUSE)	AMOUSE
('A ' *CAT MOUSE)	A MOUSE
(FAST *CAT MOUSE)	FASTMOUSE
('FAST ' *BCAT MOUSE)	FAST MOUSE
('FAST ' *TCAT MOUSE)	FASTMOUSE
('AB' *CAT 'CD')	ABCD
('AB' *BCAT 'CD')	AB CD
('AB' *TCAT 'CD')	ABCD
(%SST(&AA 1 5) *CAT (%SST(&BB 3 5))	GOOD PLACE
(%SST(&CC 1 9) *BCAT (%SST(&BB 3 5))	ALSO GOOD PLACE
(&AA *CAT ' TIME')	GOOD TIME
(&CC *BCAT TIME)	ALSO GOOD TIME

Example: Using character strings and variables

The following example shows how several character variables and character strings can be concatenated to produce a message for a work station operator. The example assumes that the variables &DAYS and &CUSNUM were declared as character variables, not decimal variables.

```
DCL      VAR(&MSG)TYPE(*CHAR)      LEN(100)
        *
        *
CHGVAR   &MSG ('Customer' *BCAT &CUSNAMD +
             *BCAT 'Account Number' *BCAT +
             &CUSNUM *BCAT 'is overdue by' +
             *BCAT &DAYS *BCAT 'days.')
```

After the appropriate variables have been substituted, the resulting message might be:

```
Customer ABC COMPANY Account Number 12345
is overdue by 4 days.
```

If the variables &DAYS and &CUSNUM had been declared as decimal variables, two other CHGVAR commands would have to change the decimal variables to character variables before the concatenation could be performed. If, for example, two character variables named &DAYSALPH and &CUSNUMALPH were also declared in the program, the CHGVAR commands would be:

```
CHGVAR  &DAYSALPH  &DAYS
CHGVAR  &CUSNUMALPH  &CUSNUM
```

Then instead of &DAYS and &CUSNUM, the new variables &DAYSALPH and &CUSNUMALPH would be specified in the CHGVAR command used to concatenate all the variables and character strings for &MSG.

For another example using character string expressions:

- “Example: Character string expressions” on page 79

Relational expressions

The operands in a relational expression can be arithmetic or character string expressions; they can also be logical constants and logical variables. Only two operands can be used with each relational operator. The data type (arithmetic, character string, or logical) must be the same for the pair of operands. The result of a relational expression is a logical value '0' or '1'.

Refer to the table under “Operators in expressions” on page 82 for the meanings of the relational operators, which can be specified by symbols (=, >, <, >=, <=, ¬ =, ¬ >, ¬ <) or their reserved values (*EQ, *GT, *LT, *GE, *LE, *NE, *NG, *NL).

If an operation involves character fields of unequal length, the shorter field is extended by blanks added to the right.

Arithmetic fields are compared algebraically; character fields are compared according to the EBCDIC collating sequence.

When logical fields are compared, a logical one ('1') is greater than logical zero ('0'). Symbolically, this is ('1' > '0').

The following are examples of relational expressions:

```
(&X *GT 25)
(&X > 25)
(&X>25)

(&NAME *EQ GSD)
(&NAME *EQ &GSD)
(&NAME *EQ 'GSD')
(&BLANK *EQ ' ')
```

Logical expressions

The operands in a logical expression consist of relational expressions, logical variables, or constants, separated by logical operators. Two or more of these types of operands can be used in combinations, making up two or more expressions within expressions, up to the maximum of five nested levels of parentheses. The result of a logical expression is a '0' or '1' that can be used as part of another expression or saved in logical variables.

The logical operators used to specify the relationship between the operands are *AND and *OR (as reserved values), and & and | (as symbols). The AND operator indicates that both operands (on either side of the operator) have to be a certain value to produce a particular result. The OR operator indicates that one or the other of its operands can determine the result.

The logical operator *NOT (or ¬) is used to negate logical variables or logical constants. All *NOT operators are evaluated before the *AND or *OR operators are evaluated. All operands that follow *NOT operators are evaluated before the logical relationship between the operands is evaluated.

The following are examples of logical expressions:

```
((&C *LT 1) *AND (&TIME *GT 1430))
(&C *LT 1 *AND &TIME *GT 1430)
((&C < 1) & (&TIME *GT 1430))
```

```
((&C<1)&(&TIME>1430))
```

```
(&A *OR *NOT &B)  
(&TOWN *EQ CHICAGO *AND &ZIP *EQ 60605)
```

Two examples of logical expressions used in the IF command are:

```
IF &A CALL PROG1  
IF (&A *OR &B) CALL PROG1
```

Operators in expressions

Operators are used in expressions to indicate an action to be performed on the operands in the expression or the relationship between the operands. There are four kinds of operators, one for each of the four types of expressions:

- Arithmetic operators (+, -, *, /)
- Character operator (||, |>, |<)
- Logical operators (&, |,)
- Relational operators (=, >, <, >=, <=, =, >, <)

Each operator must be between the operands of the expression in which it is used; for example, (&A + 4). Operators can be specified as a predefined value (for example, *EQ) or as a symbol (for example, =).

- All predefined value operators must have a blank on each side of the operator:

```
(&VAR *EQ 7)
```

- Except for the division operator (/), symbolic operators need no blanks on either side. For example, either (&VAR=7) or (&VAR = 7) is valid.

Where the division operator *follows* a variable name, the division operator must be preceded by a blank. For example, (&VAR / 5) or (&VAR /5) is valid; (&VAR/5) is not valid.

The following character combinations are the predefined values and symbols that represent the four kinds of operators; they should not be used in unquoted strings for any other purpose.

Table 11. Predefined values and symbols representing the four kinds of operators

Predefined Value	Predefined Symbol	Meaning	Type
	+	Addition	Arithmetic operator
	-	Subtraction	Arithmetic operator
	*	Multiplication	Arithmetic operator
	/	Division	Arithmetic operator
*CAT	¹	Concatenation	Character string operator
*BCAT	> ¹	Blank insertion with concatenation	Character string operator
*TCAT	< ¹	Blank truncation with concatenation	Character string operator
*AND	&	AND	Logical operator
*OR	¹	OR	Logical operator
*NOT	- ²	NOT	Logical operator
*EQ	=	Equal	Relational operator
*GT	>	Greater than	Relational operator
*LT	<	Less than	Relational operator
*GE	>=	Greater than or equal	Relational operator
*LE	<=	Less than or equal	Relational operator
*NE	- ² = ₂	Not equal	Relational operator
*NG	- ² > ₂	Not greater than	Relational operator
*NL	- ² < ₂	Not less than	Relational operator

Predefined Value	Predefined Symbol	Meaning	Type
------------------	-------------------	---------	------

Notes:

- 1 In some national character sets and in the multinational character set, the character I (hexadecimal 4F) is replaced by the character ! (exclamation point). Either ! or *OR can be used as the OR operator and either || or *CAT, !> or *BCAT, and !< or *TCAT can be used for concatenation in those character sets.
- 2 In some national character sets and in the multinational character set, the character ~ (hexadecimal 5F) is replaced by the character *. Either * or *NOT can be used as the NOT operator in those character sets.

Priority of operators when evaluating expressions

When multiple operators occur in an expression, the expression is evaluated in a specific order depending upon the operators in the expression. Parentheses can be used to change the order of expression evaluation. The following table shows the priority of all the operators used in expressions, including signed decimal values.

Priority	Operators
1	signed (+ and -) decimal values, *NOT, ~
2	*, /
3	+, - (when used between two operands)
4	*CAT, , *BCAT, !>, *TCAT, !<
5	*GT, *LT, *EQ, *GE, *LE, *NE, *NG, *NL, >, <, =, >=, <=, ~ =, ~ >, ~ <
6	*AND, &
7	*OR,

A priority of 1 is the highest priority (signed values are evaluated first); a priority of 7 is the lowest priority (OR relationships are evaluated last). When operators with different priority levels appear in an expression, operations are performed according to priorities.

When operators of the *same* priority appear in an expression, operations are performed from left to right within the expression. Parentheses can always be used to control the order in which operations are performed. The value of a parenthetical expression is determined from the innermost level to the outermost level, following the priorities stated above within matching sets of parentheses.

Built-in functions for CL

CL provides the following built-in functions:

- “%BINARY”
- “%SUBSTRING” on page 85
- “%SWITCH” on page 86

%BINARY

The binary (%BINARY) built-in function operates on a character string that is contained in a CL character variable.

%BINARY or %BIN can be used in expressions and as either operand (receiver) of the Change Variable (CHGVAR) command. See the CHGVAR command description for more information.

Note: The binary built-in function can also be used on command parameters that are defined as numeric (*DEC, *INT2 and *INT4) and EXPR(*YES) has been specified.

Sample %BINARY syntax diagram

The syntax of the binary built-in function is:

```
>>-%BINARY(-character-variable-name+-----+--)->  
                '-starting-position+-2+-'  
                '-4-'
```

```
>-----<
```

The binary built-in function treats the contents of the specified CL character variable, starting at the position specified for a length of 2 or 4 characters, as a signed binary integer.

When the binary built-in function is used with the VAR parameter on the CHGVAR command, the decimal number or arithmetic expression in the VALUE parameter is converted to a 2-byte or 4-byte signed binary integer. A decimal fraction is not included.

If the starting position and length are not specified, then a starting position of 1 and the length of the character variable specified is used. The length of the character variable must be declared as 2 or 4.

The following are examples of how the %BINARY built-in function can be used.

Example 1: Converting binary to decimal

```
DCL VAR(&N) TYPE(*DEC) LEN(3 0)  
DCL VAR(&B2) TYPE(*CHAR) LEN(2) VALUE(X'0012')  
CHGVAR &N %BINARY(&B2)
```

The content of character variable &B2 is treated as a 2-byte signed binary number and is converted to its decimal equivalent of 18. It is then assigned to the decimal variable &N.

Example 2: Converting decimal to binary

```
CHGVAR %BIN(&B2) &N
```

The number contained in the decimal variable &N is converted to a 2-byte signed binary number and is placed in the first and second bytes of the character variable &B2.

Example 3: Used within an arithmetic expression

```
CHGVAR &N VALUE(%BIN(&B2) + 4)
```

The contents of character variable &B2 is treated as a 2-byte signed binary integer and is converted to its decimal equivalent of 18. The decimal number 4 is then added and the sum, 22, is assigned to the decimal variable &N.

Example 4: Converting decimal to binary with truncation

```
CHGVAR %BINARY(&B2) VALUE(122.567)
```

The number 122.567 is truncated to the whole number 122 and is then converted to a 2-byte signed binary integer and assigned to the character variable &B2. Character variable &B2 will then contain the hexadecimal equivalent of X'007A'.

Example 5: Converting a negative number

```
DCL VAR(&B4) TYPE(*CHAR) LEN(4)  
CHGVAR %BIN(&B4) VALUE(-45)
```

The value -45 is converted to a 4-byte signed binary integer assigned to the character variable &B4. Character variable &B4 then contains the hexadecimal equivalent of X'FFFFFFD3'.

Example 6: Used on the IF command

```
IF COND(%BIN(&B4) *EQ 0) THEN(GOTO ENDIT)
```

The content of character variable &B4 is treated as a 4-byte signed binary integer and is compared to the decimal number 0. If they are equal, the command following the label ENDIT is run. If they are not equal, the command following the IF command is run.

Example 7: Varying length character string to CPP

```
PGM PARM(&P ... )

DCL VAR(&P) TYPE(*CHAR) LEN(202)

DCL VAR(&L) TYPE(*DEC) LEN(5 0)
DCL VAR(&C) TYPE(*CHAR) LEN(200)
*
*
*
CHGVAR &L %BINARY(&P 1 2)
CHGVAR &C %SST(&P 3 &L)
*
*
*
ENDPGM
```

This program is the command processing program CPP for a command with a first parameter defined with the attributes TYPE(*CHAR), LEN(200) and VARY(*YES). The first two bytes of character variable &P contain the length of the parameter as *INT2, a 2-byte signed binary integer. The character string specified on the command starts in position 3 of the variable &P. The maximum length of the character string is 200 characters.

The first CHGVAR command retrieves the length from the first two character positions of variable &P and treats the 2 bytes as a signed binary integer. The bytes are converted to the decimal equivalent of the signed binary integer, and are assigned to the decimal variable &L.

The second CHGVAR command retrieves the contents of the parameter by making variable &P a substring and assigning it to variable &C.

%SUBSTRING

The substring built-in function operates on a character string that is contained in a CL character variable or in a local data area. %SUBSTRING or %SST can be used in expressions and as either operand (receiver) of the Change Variable (CHGVAR) command. For more information, see the description of the CHGVAR command. This built-in function can be coded as either %SUBSTRING or %SST.

Sample %SUBSTRING syntax diagram

The syntax of the substring built-in function is:

```
>>-&SST(-+*LDA-----+---starting-position--length---)->
      '-character-variable-name-'
>-----<
```

This built-in function produces a substring from the contents of the specified CL character variable or local data area. The substring begins at the specified starting position in the value and continues for the length specified. For example:

```
%SST(&TEST 5 3)
```

In this example, a portion of the variable &TEST is referenced. That position (or substring) is 3 characters long and begins with the fifth character position. If &TEST contains ABCDEFGHIJ, the resulting substring will be EFG.

CL variables can also be used to specify the starting position and the length values in the function. For example:

```
CHGVAR &X %SST(*LDA &B &C)
```

The value of the character variable named &X is to be replaced by the value in the job's local data area, starting at the position obtained from variable &B and continuing for the length specified by the value in &C.

```
RTVJOBA SWS(&JOBSWS)
```

```
CHGVAR VAR(&CURSW4) VALUE(%SST(&JOBSWS 4 1))
```

In this example, the Retrieve Job Attributes (RTVJOBA) command is used to retrieve the current value of the job's eight job switches. The CHGVAR command is then used to extract the current value of the fourth job switch only and store it in the variable &CURSW4. If the value of the eight job switches retrieved in &JOBSWS is 10010000, the second 1 would be stored in &CURSW4.

%SWITCH

The built-in function %SWITCH tests one or more of the eight job switches in the current job and returns a logical value of 1 or 0. If every job switch tested by %SWITCH has the value indicated, the result is a 1 (true); if any switch tested does not have the value indicated, the result is a 0 (false).

The 8-character mask is used to indicate which job switches are tested, and what value each switch is tested for. Each position in the mask corresponds with one of the eight job switches in a job. Position 1 corresponds with job switch 1, position 2 with switch 2, and so on. Each position in the mask can be specified as one of three values: 0, 1, or X.

0 The corresponding job switch is tested for a 0 (off).

1 The corresponding job switch is tested for a 1 (on).

X The corresponding job switch is not tested.

The value in the switch does not affect the result of %SWITCH.

Sample %SWITCH syntax diagram

The syntax of the switch built-in function is:

```
>>-&SWITCH(--)->>
```

If %SWITCH(0X111XX0) is specified, job switches 1 and 8 are tested for 0s, switches 3, 4, and 5 are tested for 1s, and switches 2, 6, and 7 are not tested. If each job switch contains the value (1 or 0 only) shown in the mask, the result of %SWITCH is true (1).

Function %SWITCH can be used in the Change Variable (CHGVAR) and If (IF) commands. On the CHGVAR command, it can be used in place of a logical variable in the VALUE parameter. On the IF command, it can be used in the COND parameter as the logical expression to be tested.

The following two examples show how the same mask can be used to control a branch in a program (the IF command), or to set the value of a variable (the CHGVAR command).

```
IF COND(%SWITCH(0X111XX0)) THEN(GOTO C)
```

```
CHGVAR VAR(&A) VALUE(%SWITCH(0X111XX0))
```

If job switches 1, 3, 4, 5, and 8 respectively contain 0, 1, 1, 1, and 0 respectively when %SWITCH(0X111XX0) is specified in the IF command, the result is true and the program branches to the command having label C. If one or more of the switches tested do not have the values indicated in the mask, the result is false

and the branch does not occur. If the same mask is used in the CHGVAR command and the result is true, the variable &A is set to a '1'; if the result is false, &A is set to a '0'. Note that &A must be declared as a logical variable.


Monitoring messages

Monitorable messages are those *ESCAPE, *STATUS, and *NOTIFY messages that can be issued by each CL command that can be used in a program. You can use this information to determine which messages you want to monitor for in your program.

Using the Monitor Message (MONMSG) CL command, you can monitor for one or more messages and then specify (on the MONMSG command) what action you want taken when any of those messages are issued by the commands(s) being monitored.

You can view the monitorable messages listed at the bottom of each command description by selecting a command from the Alphabetical listing of commands.

If you have a V4R2 or later system, you can refer to the online help for an individual command to obtain its monitorable message information. To view the online command help on an iSeries 400 computer, type the command name on a command line and press F1 (Help). The error message information follows the brief description of the purpose for the command.

Refer to Chapter 8 of the CL Programming  book for information concerning messages that are sent to the QSYSMSG queue.

Command definition statements: The OS/400 control program lets users define a command that calls a program to perform some function. Users can define commands by using a command definition statement. The defined command can include the following:


- Keyword notation parameters for passing data to programs
- Default values for omitted parameters
- Parameter validity checking so the program performing the function will have correct input
- Prompt text for prompting interactive users

For additional information about command definition statements, see the following:

- “Creating user-defined commands”
- “CMD (command) statement” on page 88
- “DEP (dependent) statement” on page 89
- “ELEM (element) statement” on page 91
- “PARM (Parameter) statement” on page 105
- “PMTCTL (Prompt Control) statement” on page 123
- “QUAL (Qualifier) statement” on page 125

Creating user-defined commands

Users can define a command by entering command definition statements into a source file and running a Create Command (CRTCMD) command using the source file as input. One and only one Command (CMD) statement must be somewhere in the source file. A Parameter (PARM) statement must be provided for each parameter that appears on the command being created. If any special keyword relationships need checking, the Dependent (DEP) statement is used to define the relationships. The DEP statement can only refer to parameters that have been previously defined. These statements can appear in any order. See the

CL Programming  book for a complete description of how to use these statements to define a command.

Only one command can be defined in each source member in the source file. The CRTCMD command is run to create the command definition object from the command definition statements in one source file member. Other users can then be authorized to use the new command by the Grant Object Authority (GRTOBJAUT) command or the Edit Object Authority (EDTOBJAUT) command.

Command definition statement descriptions

The **command definition statement** of each command contains one or more of the following **command statements**:

- “CMD (command) statement”
- “DEP (dependent) statement” on page 89
- “ELEM (element) statement” on page 91
- “PARM (Parameter) statement” on page 105
- “PMTCTL (Prompt Control) statement” on page 123
- “QUAL (Qualifier) statement” on page 125

CMD (command) statement

The Command (CMD) statement specifies the prompt text for the command being created. The prompt text is displayed at a work station when a user requests prompting while entering the command that is being defined. The CMD statement can be anywhere in the source file referred to by the Create Command (CRTCMD) command; one and only one CMD statement must be used in the source file, even if no prompt text is specified for the created command. The following syntax diagram shows the syntax for the CMD definition statement.

```

      .-*NONE-----
>>-CMD--PROMPT--(--+message-identifier+---)-----<<
      '-prompt-text'-----'

```

PROMPT Parameter

Specifies the prompt text, if any, that is included in the heading (title) of the prompt display for the command being defined. The prompt text further describes the name of the command. For example, in the CRTLIB prompt heading, “Create Library (CRTLIB)”, the words *Create Library* would be specified as the prompt text in this PROMPT parameter.

Note: Prompt text for each of the *parameters* in this command can be specified in the PROMPT parameters of the PARM, ELEM, and QUAL command definition statements. They specify the prompt text for the parameters, just as the PROMPT parameter in the CMD statement specifies the prompt text for the command (in the heading).

***NONE:** No prompt text is included in the displayed heading of the prompt when the command is being prompted.

message-identifier: Specify the message identifier that specifies the message, containing no more than 30 characters, for the prompt text that is shown when the command is being prompted. If a message having the specified identifier cannot be found in the message file specified in the PMTFILE parameter of the Create Command (CRTCMD) command, the message identifier itself is used as the prompt text.

'prompt-text': Specify the prompt text that is displayed during the command prompting. It must be a character string of no more than 30 characters, enclosed in apostrophes.

Note: Variables cannot be coded for this parameter.

Example: CMD statement

```
CMD PROMPT(UCD0001)
```

This statement describes a command that is prompted with additional text in the display heading; the prompt text comes from the message identified by UCD0001.

DEP (dependent) statement

The Dependent (DEP) statement defines a required relationship between parameters and parameter values that must be checked. This relationship can refer to either the specific value of a parameter or parameters or to the required presence of parameters.

If a parameter has a default value and the parameter is not specified, the checking differs depending on whether the DEP statement is performing a specification check or a relational check. If a *specification* check is made on an unspecified parameter (checking for the presence of a value for that parameter), the system assumes that no value was specified, and the default value is *not* used. If a *relational* check is made on an unspecified parameter, the default value is used as the parameter value. The following syntax diagram shows the syntax for the DEP definition statement.

```
>>-DEP----->>
>----CTL--(--+-*ALWAYS-----+--)->
|
| (1)
| +--keyword-name-----+
| |
| | (1)
| | -&keyword-name--relational-operator-----+value-----+
| | |
| | | (2)
| | | -&keyword-name-----|
|
|
| (1) v (1) | (3)
>----PARM-----(-+--keyword-name-----+-----+>
|
| (1)
| |
| | -&keyword-name--relational-operator-----+value-----+
| | |
| | | (2)
| | | -&keyword-name-----|
|
|
|-----+-----+>
| | .-*ALL-----|
| | -NBRTRUE--(--+--relational-operator--number-true--+-+--)-|
|
|
|-----+-----+<
| | .-*NONE-----|
| | -MSGID--(--+--message-identifier--+-+--)-|
```

Notes:

1. Must not be defined as TYPE(*NULL).
2. Must not be defined as TYPE(*NULL) or PASSVAL(*NULL).
3. A maximum of 25 repetitions.

CTL Parameter

Specifies the controlling conditions that must be true before the parameter dependencies defined in the PARM statement must be true. The first keyword specified identifies the controlling parameter. The controlling condition can be specified either by a keyword name only or by a keyword name and a test relationship that determines whether the controlling condition requires the presence of the parameters it is dependent upon. The relationship between the controlling parameter and a specified value can be tested to determine whether the condition specified is met. If it is, the parameters that the controlling parameter is dependent upon must meet the requirements specified in the PARM and NBRTRUE statements.

***ALWAYS:** The parameter dependency is always checked, regardless of the form of the command.

keyword-name: Specify the keyword name of the parameter for which a value must be specified to control dependency. The keyword name is the name of the parameter that is

specified by the KWD parameter on the PARM statement defining it. If the keyword is specified, the parameter dependency is checked. The keyword must not be defined as TYPE(*NULL).

&keyword-name relational-operator value: Specify the keyword name of the controlling parameter followed by a relational operator (such as *LE or *EQ) and a value to be tested. If the tested condition is met, the parameters that the controlling parameter is dependent upon must meet the requirements specified in the PARM statement. The value must be no longer than 32 characters. The keyword must not be defined as TYPE(*NULL).

If the value being tested against has been specified as a special value or single value (SPCVAl or SNGVAL parameters of the PARM statement), the to-value must be used rather than the from-value.

The keyword name must be preceded by an ampersand (&) to indicate that the value of the keyword is tested if the relational operator and value are specified; the ampersand must not be used if the relational operator and value are not specified.

If the controlling parameter is a qualified name, the first qualifier will be used for the compare value. If the controlling parameter is a list, then the first element of the list will be used for the compare value. If the first element is a list, then the first element of that list is used, and so on.

(&keyword-name relational-operator &keyword-name): Specify the keyword name of the controlling parameter followed by a relational operator (such as *EQ) and the keyword name of another parameter whose value is compared with the value of the controlling parameter. The keywords must not be defined as TYPE(*NULL) or with PASSVAL(*NULL) specified.

PARM Parameter

Specifies the parameter dependencies that must be tested if the controlling conditions defined by the CTL parameter are true. The dependencies can be one or all of the following:

- The names of one or more parameters that will be tested to determine if they are present.
- One or more test relationships of keyword values to the values of other keywords or constants. A maximum of 25 parameters can be specified for the PARM parameter. Keywords specified in this parameter must not be defined as TYPE(*NULL).

keyword-name: Specify the keyword name of each parameter that must have a value specified for it.

&keyword-name relational-operator value: Specify the keyword name of each parameter followed by a relational operator and a value to be tested. An ampersand must precede the keyword name to indicate that the value of the keyword is tested. The value specification must be no longer than 32 characters.

If the value being used for testing comparison has been specified as a special value or single value (SPCVAl or SNGVAL parameters of the PARM statement), the to-value must be used rather than the from-value.

If the controlling parameter is a qualified name, the first qualifier will be used for the compare value. If the controlling parameter is a list, then the first element of the list will be used for the compare value. If the first element is a list, then the first element of that list is used, and so on.

&keyword-name relational-operator &keyword-name: Specify the keyword name of one parameter followed by a relational operator and the keyword name of another parameter whose value is compared with the value of the first parameter. A keyword defined with PASSVAL(*NULL) cannot be specified.

NBRTRUE Parameter

Specifies the number of parameter dependencies defined in the associated PARM statement that must be true to satisfy the control condition.

***ALL:** All the parameter dependencies specified in the PARM statement must be true to satisfy the control condition.

relational-operator number-true: Specify a relational operator and a number that specifies the number of parameter dependencies that must be true to satisfy the specified relation. For example, if (*EQ 2) is specified for NBRTRUE, then two, and only two, of the parameter dependencies must be true for the PARM statement.

Note: Variables cannot be coded for this parameter.

MSGID Parameter

Specifies the message identifier of an error message in a message file that is sent to the user if all the specified parameter dependencies have not been satisfied.

***NONE:** No special message is sent. Instead, a message generated by the command analyzer is sent to the user.

message-identifier: Specify the message identifier of the error message sent to the user if all the dependencies for this statement are not met. Messages whose identifiers begin with the 3-character prefixes *CPF* or *CPD* are retrieved from the IBM-supplied message file QCPFMSG. All other messages specified here are retrieved from the message file identified by the MSGF parameter on the CRTCMD command which is used to create the command being defined with these dependencies.

Note: Variables cannot be coded for this parameter.

Example: DEP statement

```
DEP CTL(&TYPE *EQ LIST) PARM(ELEMLIST)
```

If TYPE(LIST) is specified, the ELEMLIST parameter must be specified.

```
DEP CTL(FILE) PARM(VOL LABEL) NBRTRUE(*EQ 2)
```

If the FILE parameter is specified, both the VOL and LABEL parameters must be specified.

```
DEP CTL(GLOOP) PARM(J1 D J2) NBRTRUE(*EQ 1)
```

If the GLOOP parameter is specified, a value must be specified for one (and only one) of the J1, D, and J2 parameters.

```
DEP CTL(&LIB *EQ MYLIB)  
PARM((&PASSWORD *EQ XYZ5)  
      (&USRPRF *EQ BOBJ))  
NBRTRUE(*GE 1) MSGID(MSG001)
```

If the LIB parameter equals MYLIB, then the PASSWORD parameter must equal XYZ5, or the USRPRF parameter must equal BOBJ. Otherwise, message MSG001 will be sent to the user.

ELEM (element) statement

Element (ELEM) statements are used to define the elements of a mixed list (list elements) parameter on a command. A list parameter is a parameter that accepts multiple values that are passed together as consecutive values pointed to by a single keyword. The values are preceded by a 2-byte binary value that indicates the number of elements defined for the parameter. CL programs do not support binary values in variables, but you can use the %BINARY built-in function to examine the length.

A list element is the value that represents one value among a group of values organized in a specific order in a list. If all of the list elements are not of the same type, one ELEM statement must be used for each element that appears in the list being defined. If all the elements are of the same type (a simple list), individual ELEM statements are not required. For a simple list, specify the number of elements in the list on the MAX parameter of the PARM statement.


```

|          .-0-----|
|-MIN(--+minimum-number+--)-|
>-----+----->
|          .-1-----|
|-MAX(--+maximum-number+--)-|
>-----+----->
|          .-*YES--|
|-ALWUNPRT(--+*NO+--)-|
>-----+----->
|          .-*YES--|          .-*NO--|
|-ALWVAR(--+*NO+--)-|          -PGM(--+*YES+--)-|
>-----+----->
|          .-*NO--|          .-*NO----|
|-DTAARA(--+*YES+--)-|          -FILE(--+*IN+--)-|
|                                     +-*OUT----+
|                                     +-*UPD----+
|                                     +-*INOUT--+
|                                     '-*UNSPFD-'
>-----+----->
|          .-*NO--|          .-*NO--|
|-FULL(--+*YES+--)-|          -EXPR(--+*YES+--)-|
>-----+----->
|          .-*NO-----|          .-*INT2--|
|-VARY(--+*YES+--)-|          +*INT4+--+--)-|
|          (2)
>-----+----->
|          .-*NO--|
|-PASSATR(--+*YES+--)-|
>-----+----->
|          .-*MONO-----|
|-CASE(--+*MIXED+--)-|
|          (3)
>-----+----->
|          .-*YES----|
|-DSPINPUT(--+*PROMPT+--)-|
|          '-*NO-----'
>-----+----->
|          .-*VALUES-----|
|-CHOICE(--+*NONE+--)-|
|          +-*PGM-----+
|          +message-identifier+
|          '-choices-text'-----|
>-----+----->
|          .-*NONE-----|          .-*LIBL/-----|
|-CHOICEPGM(--+*LIBL/-----+--program-name+--)-|
|          +*CURLIB/-----+
|          '-library-name/'-----|
>-----+----->
|          .-*CALC-----|
|-INLPMPTLEN(--+*PWD+--)-|
|          '-initial-prompt-length-'

```

```

>-----+-----+-----+-----+-----+----->
|          .-*NONE-----|
|'-PROMPT(--+message-identifier-+---)--'|
|          '-prompt-text'-----'|

```

Notes:

1. A maximum of 300 repetitions.
2. *YES is valid only for the following parameter types: *CHAR, *NAME, *SNAME, *CNAME, *PNAME, *GENERIC, *LGL, *VARNAME, *CMD, *CMDSTR, and *X.
3. This value can be specified only for *CHAR and *PNAME parameter types.

TYPE Parameter

Specifies the type of list element being defined. The element can be an integer, a decimal or logical value, or a quoted or unquoted character string that can be a name, label, date, or time. Enter one of the following values to specify the type of element.

Note: For more information on names (below), see “Object naming rules” on page 141.

***NAME:** The list element is a character string that represents a basic name. The maximum length of the name is 256 characters; the first character must be A-Z, \$, @, or #. The remaining characters can be the same as the first character but can also include the numbers 0 through 9, underscores (_), and periods (.). The name can also be a string of characters starting and ending with double quotation marks (“”). If a special value is used (as in *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***SNAME:** The list element is a character string that represents a simple name. The maximum length of the name is 256 characters; the first character must be A-Z, \$, @, or #. The remaining characters can be the same as the first but may also include the numbers 0 through 9 and underscores (_). Periods (.) are not allowed. If a special value is used (such as *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***CNAME:** The list element is a character string that represents a communications name. The maximum length of the name is 256 characters; the first character must be A-Z, \$, @, or #. The remaining characters can be the same as the first character but may also include the numbers 0 through 9. Periods (.) and underscores (_) are not allowed. If a special value is used (such as *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***PNAME:** The list element is a character string that represents a path name. The path name can be enclosed in apostrophes. If the path name contains any special characters (not including an asterisk (*)), it must be enclosed in apostrophes. The maximum length of the path name character string is 5000 characters.

Path names are used in the integrated file system. See the information in the Integrated File System section for more information on using the list element *PNAME.

***GENERIC:** The list element is a character string that represents a generic name. A generic name contains a maximum of 255 characters followed by an asterisk (*) or 256 characters without an asterisk and must conform to the rules for “Generic object names” on page 140. The name identifies a group of objects whose names all begin with the characters preceding the *. If an * is not included, the system assumes that the generic name is a complete object name.

***DATE:** The list element is a character string that represents a date. When entering the command, the year may be specified in either 2 digits or 4 digits. If a 2-digit year is specified, the date is assumed to be in the range of January 1, 1940 through December 31, 2039. If a 4-digit year is specified, the date may be in the range of August 24, 1928 through May 9, 2071. When the date is passed to the CPP, it is always passed in the format *Cyyymmdd*, where C = century, yy = year, mm = month, and dd = day. The century digit is set to 0 for years 19xx, and it is set to 1 (one) for years 20xx. When a date value is specified in this ELEM statement,

it must be specified without quotation marks in one of the following formats: *mmddy*, *mmddy*, or *Cyymmdd*. If the user enters a date when the command is run, it must be entered in the job date format. The job date separator specifies the optional separator character that must be used when the date is entered. If the separator character is used, the date must be enclosed in apostrophes.

***TIME:** The list element is a character string that represents a time. It is in the format *hhmmss*, where hh = hours, mm = minutes, and ss = seconds. It is passed to the CPP in a 6-byte character string as *hhmmss*. Values specified in this statement must be in the format *hhmmss*. The job time separator specifies the optional separator character that can be used when the time is entered. If the separator character is used, the time must be enclosed in apostrophes.

***CHAR:** The list element is a character string that optionally can be enclosed in apostrophes. If the character string contains any special characters (not including an asterisk (*)), it must be enclosed in apostrophes. The maximum length of the character string is 5000 characters.

***DEC:** The list element is a packed decimal number.

***LGL:** The list element is a logical value, either a one ('1') or a zero ('0').

***HEX:** The list element value is in hexadecimal form. The specified characters must be 0 through F. They are converted to hexadecimal (EBCDIC) characters (2 hex digits per byte), right justified, and padded with zeros. If the value is enclosed in apostrophes, an even number of digits is required. If the value is not enclosed in apostrophes, the number of digits may be odd or even.

***ZEROELEM:** The list element is always considered to be a list of zero elements for which no value can be specified in the command. It is used to prevent a value from being entered as an element in a list even though the CPP expects one. An element for which *ZEROELEM is specified is not prompted, although the other elements in the parameter are prompted and are passed to the CPP as a list.

***INT2:** The list element is an integer that is passed as a 2-byte signed binary number. CL programs do not support using binary values in variables.

***INT4:** The list element is an integer that is passed as a 4-byte signed binary number. CL programs do not support using binary values in variables.

***UINT2:** The list element is an integer that is passed as a 2-byte unsigned binary number. CL programs do not support using binary values in variables.

***UINT4:** The list element is an integer that is passed as a 4-byte unsigned binary number. CL programs do not support using binary values in variables.

***VARNAME:** (For IBM-supplied commands) The list element is a variable name that is passed as a character string. The name can contain a maximum of 11 characters, including the initial ampersand (&).

statement-label: Specify a qualified name or a mixed list of values. The statement label specified here by the TYPE parameter is the statement label that identifies the first of a series of QUAL or ELEM statements that further describe the qualified name or the mixed list being defined. The label must be the same as the label specified by TYPE(statement-label) on the PARM statement for this list.

LEN Parameter

Specifies the length of the list element value that is passed to the CPP. If TYPE was specified as *INT2, *INT4, *UINT2, *UINT4, *DATE, *TIME, *ZEROELEM, or statement-label, LEN is not allowed. If TYPE(*DEC) was specified, the decimal length is specified in the form (n1 n2), where n1 specifies the total number of digits in the value (including the decimal portion), and n2 specifies the number of allowable decimal digits to the right of the decimal point. The value for n2 is optional, and zero is assumed if n2 is not entered. If TYPE is other than *DEC, the decimal portion (n2) must be omitted and only the number of characters must be specified.

Table 12. Lengths of Data Types That Can Be Coded

Data Type	Default Length	Maximum Length ¹
*DEC	(15 5)	(24 9)
*LGL	1	1
*CHAR	32	5000
*NAME	10	256
*SNAME	10	256
*CNAME	10	256
*PNAME	32	5000
*GENERIC	10	256
*HEX	1	256
*VARNAME	11	11

Note:

¹ The maximum length shown here is the maximum length allowed by the Command Definition. The high-level language used as the CPP for the command may have different maximum lengths for these data types (for example, *DEC values in CL programs have a maximum length of (15 9)).

Whereas the maximum shown here is the maximum for the values used in the command when it is run, the maximum length of character constants specified in the command definition is limited to 32 characters. This restriction applies to the following parameters: CONSTANT, DFT, VALUES, REL, RANGE, SPCVAL, and SNGVAL.

For data types whose length cannot be coded, the following are the maximum lengths and the lengths passed.

Table 13. Lengths of Data Types That Cannot Be Coded

Data Type	Maximum Length	Length Passed
*DATE ¹	8	7
*TIME ²	8	6
*ZEROELEM	0	0
*INT2 ³	6	2
*INT4 ⁴	11	4
*UINT2 ⁵	6	2
*UINT4 ⁶	11	4
statement-label ⁵	—	—

Notes:

- ¹ If a date is specified, the value is passed as 7 characters.
- ² If a time is specified, the value is passed as 6 characters.
- ³ The value must meet the following condition: $-2^{15} \leq \text{value} \leq 2^{15}-1$. The value is passed as a 2-byte signed binary number.
- ⁴ The value must meet the following condition: $-2^{31} \leq \text{value} \leq 2^{31}-1$. The value is passed as a 4-byte signed binary number.
- ⁵ The value must meet the following condition: $0 \leq \text{value} \leq 2^{16}-1$. The value is passed as a 2-byte unsigned binary number.
- ⁶ The value must meet the following condition: $0 \leq \text{value} \leq 2^{32}-1$. The value is passed as a 4-byte unsigned binary number.
- ⁷ The length of the data accepted and passed is defined by the ELEM or QUAL statement that the label identifies.

CONSTANT Parameter

Specifies that a value is passed to the CPP as a constant for the list element when the

command being defined is processed. The value does not appear externally on the command. If specified, the value must satisfy the requirements specified by the TYPE, LEN, VALUES, REL, RANGE, SPCVAL, and FULL parameters. As noted in the LEN parameter chart, if a character constant is specified in this parameter, it can be no longer than 32 characters. CONSTANT is not valid with TYPE(*ZEROELEM), EXPR(*YES), or MAX(>1), or if DFT was coded for ELEM.

If a constant is specified for the element being defined, no prompt text can be specified for the PROMPT parameter of this ELEM statement. However, the other elements of the list parameter (of which this list element is a part) are still prompted, and their values (along with this constant value) are still passed to the CPP as a list.

Variables cannot be coded for this parameter.

RSTD Parameter

Specifies whether the value entered for the list element (specified in the ELEM statement) is restricted to only one of the values given in the VALUES, SPCVAL, or SNGVAL parameters, or whether the value can be any value that satisfies the requirements specified by the TYPE, LEN, REL, RANGE, SPCVAL, SNGVAL, and FULL parameters.

***NO:** The value entered for the list element defined by this ELEM statement can be anything that matches the requirements specified by the TYPE, LEN, REL, RANGE, SPCVAL, SNGVAL, and FULL parameters in this ELEM statement.

***YES:** The value entered for the list element in this ELEM statement is restricted to one of the values in the VALUES parameter, or to one of the from-values in the SPCVAL or SNGVAL parameter. *YES cannot be specified if TYPE(statement-label) or TYPE(*ZEROELEM) is specified.

DFT Parameter

Specifies the default value that is assigned to the list element. That is, the default value is used as the value of the list element if the user omits the parameter that represents this list element, or specifies *N for the element, while coding or entering the command. The default value must satisfy one of the following:

- It must match the element requirements specified by the TYPE, LEN, REL, RANGE, and FULL parameters.
- It must be one of the from-values in the SPCVAL or SNGVAL parameters.
- If the default is a character constant, it can have no more than 32 characters (as noted in the LEN parameter chart).
- If RSTD(*YES) is specified, it must be in the list of values in the VALUES parameter or in the list of from-values of the SPCVAL or SNGVAL parameters.
- If this ELEM statement itself defines a list, the default value must be specified in the SNGVAL parameter.

The DFT parameter is valid only if MIN is 0, which means the element defined by this ELEM statement for this list is optional. The DFT parameter is not valid if the CONSTANT parameter is specified. A default cannot be specified if TYPE(*ZEROELEM) is specified; in that case, an assumed default is passed.

If DFT is not specified, the default assumed is as follows, depending on the specified element type:

Assumed Default	Element Types
<i>0</i>	*DEC *INT2 *INT4 *UINT2 *UINT4 *ZEROELEM
<i>'0'</i>	*LGL
<i>zeros</i>	*DATE *TIME *HEX
<i>blanks</i>	*CHAR *NAME *SNAME *CNAME *PNAME *GENERIC *VARNAME

An *assumed* default value is not displayed by the command prompt; a blank input field is shown instead. If a default is specified in the DFT parameter, it is displayed by the prompt exactly as specified.

value: Specify the default value that meets the specified requirements or that is one of the values specified in the SPCVAL, SNGVAL, or VALUES parameters.

Variables cannot be coded for this value.

VALUES Parameter

Specifies a list of up to 300 constants (fixed values) from which one constant can be specified as the value of the list element. The VALUES parameter is valid only if all of the following are true: RSTD(*YES) is specified, both RANGE and REL are *not* specified, and each constant matches the attributes specified by the TYPE, LEN, and FULL parameters in this ELEM statement. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. Enter the constants (not more than 300) that can be specified as the value of the list element. The VALUES parameter is not valid for TYPE(statement-label) or for TYPE(*ZEROELEM).

If this ELEM statement is defining the first element in a list, the value specified for this parameter cannot be the same as the value specified in the SNGVAL parameter on either the PARM or ELEM statement that points to this ELEM statement.

REL Parameter

Specifies the relationship between the list element value and a constant value.

To specify the relationship, enter one of the following relational operators followed by a constant.

*LT	less than
*LE	less than or equal to
*EQ	equal to
*GE	greater than or equal to
*GT	greater than
*NL	not less than
*NE	not equal to
*NG	not greater than

The REL parameter is not valid if TYPE is specified as *LGL, *VARNAME, *ZEROELEM, or statement label; or if either RANGE or VALUES is specified.

If a character type is specified by TYPE(*CHAR), the EBCDIC value of the character string is used as an unsigned integer in the comparison. As noted in the LEN parameter chart, a character constant specified in this parameter can be no longer than 32 characters.

RANGE Parameter

Specifies the range (the limits) for the value of the list element. The list element value must be greater than or equal to the lower limit value specified, and it must be less than or equal to the upper limit value specified. The value tested is the value sent to the CPP, not the user-specified value. For example, 15 would be valid if RANGE was specified as (0 16).

For nonnumerical data types, such as *CHAR, the range of values and the data specified will be left-justified and padded on the right with blanks. A numeric range should not be used to define an interval for nonnumerical data unless leading zeros are specified or the data is only 1 character in length.

The RANGE parameter is not valid if either REL or VALUES is specified, or if TYPE is specified as *LGL, *VARNAME, *ZEROELEM, or statement label. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters.

SPCVAL Parameter

Specifies a list of up to 300 entries that define special values that can be entered for the element defined by this ELEM statement. Each entry specifies a character string (a from-value) that can be entered even though it may not meet all validity checking requirements. If the entered character string matches the from-value of one of the entries, and the to-value is specified, the string is replaced with the to-value and is then passed to the CPP without further checking. If the to-value is omitted, the from-value is passed to the CPP. The SPCVAL parameter is not valid for TYPE(statement-label) or for TYPE(*ZEROELEM).

If a to-value of *CURLIB is specified, the name of the current job library is passed to the CPP instead of the value *CURLIB. If the from-value is *CURLIB and no to-value is specified, or if the to-value is *CURLIB and it is enclosed in apostrophes, the value *CURLIB is passed to the CPP.

The from-value is a character string, but the to-value can be anything that is passable. However, for TYPE(*DATE), the to-value must be specified unquoted in *mmdyy*, *mmdyyyy*, or *Cyymmdd* format. If a CL variable is used for the from-value, its type must be *CHAR. If this ELEM statement is defining the first element in a list, the value specified for the from-value cannot be the same as the value specified in the SNGVAL parameter on either the PARM or ELEM statement that points to this ELEM statement.

The to-value must be no longer than LEN specifies. If TYPE is *DEC, *INT2, *INT4, *UINT2, or *UINT4, the type of the to-value must be the same. If TYPE is a character type (such as *CHAR, *LGL, or *DATE), the to-value must be a character string. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. If a to-value is not specified, the from-value must be passable.

Variables cannot be coded for this element.

SNGVAL Parameter

Specifies a list of up to 300 single values that can be specified for an element being defined as a statement label, or that is to have two or more list elements (defined by the MAX parameter) in its nested list. Any one of the single values can be used instead of a nested list of values or a qualified name that the element is defined to accept. Each entry specifies a character string (a from-value) that can be entered. If an entered character string matches the from-value of one of the entries and the to-value is specified, the data is replaced with the to-value and is then passed to the CPP without further checking. If the to-value is omitted, the from-value is passed to the CPP.

If this ELEM statement defines the first element in a list, the value specified for the from-value cannot be the same as the value specified in the SNGVAL parameter on either the PARM or ELEM statement that points to this ELEM statement.

The to-value (or the from-value, if the to-value is omitted) must be passable, as specified in the SPCVAL parameter. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. SNGVAL can be specified only if MAX is greater than one or TYPE is specified as a statement

label; it is not valid for TYPE(*ZEROELEM). Each single value can only substitute for a list of values or a qualified name; it cannot be a list element or qualifier. It is passed as the first element of the list.

If a to-value of *CURLIB is specified, the name of the current job library is passed to the CPP instead of the value *CURLIB. If the from-value is *CURLIB and no to-value is specified, or if the to-value is *CURLIB and it is enclosed in apostrophes, the value *CURLIB is passed to the CPP.

Variables cannot be coded for this element.

MIN Parameter

Specifies the minimum number of values that must be entered for the list element being defined.

For an element that does not allow multiple like values, only zero (0) for optional and one (1) for required can be specified as the minimum number of values.

For an element that allows multiple like values because a value greater than one (1) is specified in the MAX parameter, zero (0) indicates that no values must be entered; therefore, it is an *optional* element. A value equal to or greater than one (1) indicates the minimum number of values that must be entered for the element. Therefore, it is a *required* element. The value specified for MIN cannot exceed the value specified for the MAX parameter.

The number specified tells how many list elements are required in another list. If MIN is not specified, zero (0) is assumed, meaning that the element is optional.

0: The list element is optional; it does not have to be entered.

minimum-number: Specify the minimum number of elements that must be specified in the nested list. If 1 is the assigned value, it specifies that at least one value is required for the element. If a number greater than one (1) is specified, the element contains a list that must have at least as many elements as the number specified.

MAX Parameter

Specifies, if this ELEM statement is defining a simple list element, the maximum number of elements that this list element can have in its nested list. If a value greater than 1 is specified, the element is capable of accepting multiple like values (that is, a simple nested list). All values entered for this element (at the time the command is run) must satisfy the validity checking requirements specified by the values specified in the other parameters on this ELEM statement.

Note: The values for a nested list are passed consecutively, preceded by a 2-byte binary value that indicates the number of values entered in the list element by the user. CL programs do not support the handling of binary values in variables, but you can use the %BINARY built-in function to examine the length.

1: The list element accepts only one value; there is no nested list.

maximum-number: Specify the maximum number of elements that the list element can accept. The specified maximum must be greater than or equal to the value specified in MIN, and less than or equal to 300. If the maximum is greater than one (1) and TYPE is not a statement label that identifies a QUAL statement or another ELEM statement, the parameter-which is also an element -is a simple list of like values (that is, each element in the list has the same requirements, such as type and length). If TYPE(statement-label) is specified and it points to the label of a QUAL statement or another ELEM statement, MAX should only be specified greater than 1 if a list of lists or a list of qualified names is to be accepted. A maximum greater than one (1) is not valid if the CONSTANT parameter is also specified.

ALWUNPRT Parameter

Specifies whether this ELEM statement should accept the hexadecimal characters X'FF' or those in the range of X'00' to X'3F'. This parameter is valid only for TYPE(*CHAR) or TYPE(*X).

***YES:** Allows any characters to be sent to the display or printer.

***NO:** Does not allow unprintable characters to be passed to the command processing program.

ALWVAR Parameter

Specifies whether to allow variable names for the element. If TYPE was specified as *VARNAME, *ZEROELEM, *NULL, or statement-label, ALWVAR(*NO) is not allowed.

***YES:** Allows the use of variable names for the element.

***NO:** Does not allow the use of variable names for the element.

PGM Parameter

Specifies whether the list element is a program name. PGM(*YES) is valid only for a statement-label, *CHAR, *NAME, *SNAME, *CNAME, and *GENERIC types. The specification of the PGM(*YES) parameter has no effect on the element being defined by the ELEM statement; it only indicates to the compiler that the value for this element is a program name. This information is stored so it can be included in the output of the Display Program References (DSPPGMREF) command.

***NO:** The element defined in this ELEM statement is not a program name.

***YES:** The element is a program name.

DTAARA Parameter

Specifies whether the list element is a data area name. DTAARA(*YES) is valid only for statement-label, *CHAR, *NAME, *SNAME, *CNAME, and *GENERIC types. The specification of the DTAARA(*YES) parameter has no effect on the element being defined by the ELEM statement; it only indicates to the CL compiler that the value for this element is a data area. This information is stored so it can be included in the output of the Display Program References (DSPPGMREF) command.

***NO:** The element defined in this ELEM statement is not a data area name.

***YES:** The element is a data area name.

FILE Parameter

Specifies whether the list element is a file name and the expected use of the file. The element can be specified as the name of a file that has a specific use so that, at CL compile time, the names can be used to get file reference information about where the files are used. FILE is valid only if the value for TYPE is statement-label, *CHAR, *NAME, *SNAME, *CNAME, or *GENERIC. The specification in the FILE parameter has no effect on the list element being defined by the ELEM statement; it only indicates to the CL compiler that the value for this element is a file name and what type of file it is. This information is stored so it can be included in the output of the DSPPGMREF (Display Program References) command. One of the following types of files can be specified:

***NO:** The list element defined in this ELEM statement is not a file name.

***IN:** The list element is an input file name.

***OUT:** The list element is an output file name.

***UPD:** The list element is an update file name.

***INOUT:** The list element value is the name of a file to be used for both input and output.

***UNSPFD:** The list element value is the name of a file, but its use cannot be specified.

FULL Parameter

Specifies whether the number of characters in the list element must be exactly the same as the number specified in the LEN parameter (if specified) or its default length (if LEN is not specified).

***NO:** The number of characters in the list element can be less than that specified by the LEN parameter.

***YES:** The number of characters in the list element must equal the number specified by the LEN parameter or the default length for that type. The exact length is valid only for element types *LGL, *CHAR, *NAME, *SNAME, *CNAME, *GENERIC, *VARNAME, and *HEX.

EXPR Parameter

Specifies whether the list element can accept an expression containing a characters that are combined or linked together (concatenation). Valid character concatenation operators are as follows:

Concatenation	*CAT or,
Blank insertion with concatenation	*BCAT or, >
Blank truncation with concatenation	*TCAT or, <

***NO:** The element value cannot be a concatenation expression.

***YES:** The element value can be a concatenation expression. *YES is not valid if a value is specified for the CONSTANT parameter.

VARY Parameter

Specifies whether the list element value passed to the CPP is preceded by a length value that indicates the number of characters entered for the element's value. This parameter may be specified as a single value (*NO) or as a list of two values (elements).

***NO:** The element value is not preceded by a length value.

Element 1: Return Length Value

***YES:** The element value passed to the CPP is preceded by a field that indicates the number of characters actually specified for the parameter. *YES is valid only for the following parameter types: *CHAR, *NAME, *SNAME, *CNAME, *PNAME, *GENERIC, *LGL, *VARNAME, *CMD, *CMDSTR, and *X.

Note: The length value is the actual number of characters entered for the command parameter with trailing blanks removed. The length value passed may be different than the defined parameter length or the declared variable length. The length of the field containing the character string data is determined by the defined length for the parameter or the declared LEN for CL Program variables. The length value defines how many characters in the character string data field were actually entered for the command parameter.

Element 2: Value Length

***INT2:** The length of the element value is an integer passed as a 2-byte signed binary number.

***INT4:** The length of the element value is an integer passed as a 4-byte signed binary number.

PASSATR Parameter

(For IBM-supplied commands) Specifies whether an attribute byte is to be passed to the CPP with the list element data. PASSATR is not valid for TYPE(statement-label) or for TYPE(*ZEROELEM).

***NO:** No attribute byte is passed with the list element.

***YES:** An attribute byte is passed with the list element; the attribute byte indicates whether the data value came from the default, the data type of the value, and (if TYPE(*CHAR) was specified) whether the character string was enclosed in apostrophes.

CASE Parameter

Specifies whether the value that is passed to the CPP is changed from lowercase to uppercase, or is preserved in the case specified on the command parameter.

***MONO:** The element value is changed from lowercase to uppercase. Single quoted parameters with apostrophes preserve the case whether or not this value is specified.

***MIXED:** The element value is preserved in the case specified on the command parameter. This value can be specified only for *CHAR and *PNAME parameter types.

DSPINPUT Parameter

Specifies whether the keyword value is shown (displayed) in the job log or on a prompt display. DSPINPUT(*PROMPT) and DSPINPUT(*NO) are valid only for the following element types: *CHAR, *NAME, *SNAME, *CNAME, *GENERIC, *DEC, *LGL, *INT2, *INT4, *UINT2, *UINT4, *DATE, *TIME, and *HEX.

Note: The DSPINPUT parameter will have no effect on the job log entries for a database reader job, for imbedded commands (for example, a command submitted on the SBMJOB command), or for commands executed using the QCMDXEC or QCAEXEC APIs.

***YES:** The element value is shown both on the prompt screen and in the job log.

***PROMPT:** The element value is shown on the prompt screen but not in the job log.

***NO:** The element value is not shown in either the job log or a prompt screen. When a previously entered command is retrieved, the nondisplayed field entries must be retyped and their previous values are not retrievable. When a job log entry is created, the nondisplayed field is replaced by empty parentheses ().

CHOICE Parameter

Specifies the text that is displayed to the right of the prompt line of each parameter on the prompt screen. Up to 30 characters of text can be displayed.

***VALUES:** Each possible value is displayed in the possible values field, separated by a comma and a space. If values are specified for the Default, Single value, or Special value parameters, the first value displayed is the default value; the next value is a single value, and the values following that are special values. If there are too many values to fit in 30 characters, the last value is followed by three periods.

Examples of possible values text follow:

- If *NO is specified on the RSTD parameter and *DEC is specified on the TYPE parameter and the RANGE parameter is not specified, the word "Number" is displayed in the possible values field. The resulting line will appear in the form: Number, *XXX, *YYY, *ZZZ...
- If *NO is specified on the RSTD parameter and *DEC is specified on the TYPE parameter and the RANGE parameter is specified, the range of possible values is displayed in the possible values field. The resulting line will appear in the form: a-b, *XXX, *YYY, *ZZZ (where a and b are numerals defining the range).

- If *YES is specified on the RSTD parameter, the possible values displayed are determined by the VALUES parameter, the SNGVAL parameter, and the SPCVAL parameter. The resulting line will appear in the form: *XXX, *YYY, *ZZZ...

***NONE:** No values are displayed.

***PGM:** A program that is called determines the values that are displayed. The program that is called is identified in CHOICEPGM parameter.

message-identifier: Specify the message ID of the message used to retrieve the message containing the text for the possible values field. The message file specified on the PMTFILE parameter of the Create Command (CRTCMD) command is used to find the message.

'choices-text': Specify no more than 30 characters, enclosed in apostrophes.

CHOICEPGM Parameter

Specifies the name of the program that is called during the prompting to fill in the possible choices text and the permissible values during prompting. This parameter must be specified only if CHOICE(*PGM) is specified.

***NONE:** No program is identified to fill in the possible choices text and permissible values.

The possible library values are:

***LIBL:** The library list is used to locate the program name.

***CURLIB:** The current library for the job is used to locate the program name. 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 where the program name is located.

program-name: Specify the name of the program to be called during prompting to fill in the possible choices text or to supply a permissible value.

If any exception occurs when the program is called, no possible-choices text will be left blank, and the list of permissible values will be obtained from the command.

INLPMTLEN Parameter

Specifies the length of the input field initially displayed for the element when the command is prompted. The user can extend the field to a maximum length of 512 bytes by entering an ampersand (&) in the first position of the field, followed by a blank. INLPMTLEN is valid only if TYPE is specified as *CHAR, *NAME, *SNAME, *CNAME, *PNAME, *GENERIC, or *HEX. If FULL(*YES), RSTD(*YES), or CONSTANT are specified, INLPMTLEN(*CALC) must be specified or defaulted.

***CALC:** The prompter will determine the length of the prompt field based on the type and length of the parameter.

initial-prompt-length: Specify the initial length in bytes of the prompt field. Valid values are 1-12, 17, 25, 32, 50, 80, 132, 256, and 512.

PROMPT Parameter

Specifies the prompt text, if any, that is used for the list element being defined in this ELEM statement. The prompt text describes the element to the user, who may enter a response to the information displayed. Prompt text cannot be specified if TYPE(*ZEROELEM) is specified or if a constant value is specified in the CONSTANT parameter.

***NONE:** No prompt text is displayed for the list element defined by this ELEM statement. The input field still asks for this list element, but no text is displayed with it.

message-identifier: Specify the message identifier that identifies the prompt text message of up to 30 characters displayed when the program is asking for the list element. If a message having the specified identifier cannot be found in the message file contained in the PMTFILE parameter of the Create Command (CRTCMD) command, the message identifier itself is used as the prompt text.

'prompt-text': Specify the prompt text displayed when the program is asking for the list element. The text must be a character string of no more than 30 characters, enclosed in apostrophes.

Example: ELEM statement

Example 1: Defining a JOBDESC Parameter

```
>>-+-----+-----<<
      '-JOBDESC(-job-name--number---)-'
```

Command definition statements:

```
      PARM KWD(JOBDESC) TYPE(L1)
L1:    ELEM TYPE(*NAME) MIN(1)
      ELEM TYPE(*DEC) LEN(2) MIN(1) REL(*LE 60)
```

The parameter named JOBDESC can be omitted, but, if present, it *must* be a list of two elements. The first element is a name, and the second is a 2-digit number that is less than or equal to 60.

Example 2: Defining a RANGE Parameter

```
      .-*SAME-----
>>-RANGE(-+-lower-value--upper-value--+-)------<<
```

Command definition statements:

```
      PARM KWD(RANGE) TYPE(L1) DFT(*SAME) +
      SNGVAL((*SAME 101))
L1:    ELEM TYPE(*DEC) MIN(1) REL(*LE 100)
      ELEM TYPE(*DEC) MIN(1) REL(*LE 100)
```

The parameter named RANGE can be omitted, but, if present, it must be a list of two numbers, neither of which can be greater than 100. Validity checking is performed to determine that the lower-limit value is less than the upper-limit value. To allow the CPP to determine whether the value passed is a user-specified value or the default, *SAME is mapped to 101.

PARM (Parameter) statement

The Parameter (PARM) statement defines a parameter of a command being created. A parameter is the means by which a value is passed to the command processing program (CPP). One PARM statement must be used for each parameter that appears in the command being defined. The order in which the PARM statements are entered into the source file determines the order in which the parameters must be specified when the command is entered in positional form and the order in which they are passed to the


```

| .-*VALUES-----|
|-CHOICE--(--+*NONE-----)|
| +-*PGM-----+
| +-message-identifier-+
| '-choices-text'-----|
>-----+----->
| .-*NONE-----|
| .-*LIBL/-----|
|-CHOICEPGM--(--+-----program-name-----)|
| +-*CURLIB/-----+
| '-library-name/-'|
>-----+----->
| .-*NONE-----|
| .-*PMTRQS-----|
|-PMTCTL--(--+-----)|
| '-statement-label-|
>-----+----->
| .-*NONE-----|
| .-*LIBL/-----|
|-PMTCTLPGM--(--+-----program-name-----)|
| +-*CURLIB/-----+
| '-library-name/-'|
>-----+----->
| .-*NO--|
|-KEYPARM--(--+*YES-+---)|
>-----+----->
| .-*CALC-----|
| .-*PWD-----|
|-INLPMTLEN--(--+-----)|
| '-initial-prompt-length-|
>-----+----->
| .-*NONE-----|
|-PROMPT--(--+-----message-identifier-----relative-prompt-number-----)|
| '-prompt-text'-----|
>-----+----->

```

Notes:

1. A maximum of 300 repetitions.
2. *YES is valid only for the following parameter types: *CHAR, *NAME, *SNAME, *CNAME, *PNAME, *GENERIC, *LGL, *VARNAME, *CMD, *CMDSTR, and *X. *YES must be specified if PASSATR(*YES) and RTNVAL(*YES) are specified.
3. This value can be specified only on *CHAR and *PNAME parameter types.

KWD Parameter

Specifies the keyword name of the parameter being created. The keyword identifies the parameter in the command, and it is used when the parameter is entered in keyword form. Enter the keyword name using no more than 10 alphanumeric characters, the first character being alphabetic.

TYPE Parameter

Specifies the type of value being defined. The value can be an integer, a decimal or logical value, or a quoted or unquoted character string that can be a name, label, date, or time. Enter one of the following values to specify the type of parameter.

Note: For more information on names (below), see “Path names (*PNAME)” on page 143.

***NAME:** The parameter value is a character string that represents a basic name. The maximum length of the name is 256 characters; the first character must A-Z, \$, @, or #. The remaining characters can be the same as the first character but can also include the numbers 0 through 9,

underscores (_), and periods (.). The name can also be a string of characters starting and ending with double quotation marks ("). If a special value is used (as in *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***SNAME:** The parameter value element is a character string that represents a simple name. The maximum length of the name is 256 characters; the first character must be A-Z, \$, @, or #. The remaining characters can be the same as the first but may also include the numbers 0 through 9 and underscores (_). Periods (.) are not allowed. If a special value is used (such as *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***CNAME:** The parameter value is a character string that represents a communications name. The maximum length of the name is 256 characters; the first character must be A-Z, \$, @, or #. The remaining characters can be the same as the first character but may also include the numbers 0 through 9. Periods (.) and underscores (_) are not allowed. If a special value is used (such as *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***PNAME:** The parameter value is a character string that represents a path name. The path name can be enclosed in apostrophes. If the path name contains any special characters (not including an asterisk (*)), it must be enclosed in apostrophes. The maximum length of the path name character string is 5000 characters.

Path names are used in the integrated file system. For more information on using the parameter value *PNAME, see the Integrated File System topic.

***GENERIC:** The parameter value is a character string that represents a generic name. A generic name contains a maximum of 255 characters followed by an asterisk (*) or 256 characters without an asterisk and must conform to the rules for generic names. The name identifies a group of objects whose names all begin with the characters preceding the *. If an * is not included, the system assumes that the generic name is a complete object name.

***CMDSTR:** The parameter value is a command string that is checked for validity by the command analyzer. It is passed to the CPP as a character string.

The command analyzer rebuilds the command string when it checks it for validity. When the command is rebuilt, keywords are added to parameters that were specified positionally, parameters can be reordered, and parameters that contain characters that cannot be printed (X'FF' and X'00 - X'3F') are converted to hexadecimal notation. As a result, the rebuilt command string may be substantially longer than the original command string. If the length of the rebuilt command is longer than the allowed length specified with the LEN keyword, the command will fail.

Note: Selective prompting is not allowed with the *CMDSTR parameter.

***DATE:** The parameter value is a character string that represents a date. When entering the command, the year may be specified with either 2 digits or 4 digits. If a 2-digit year is specified, the date is assumed to be in the range of January 1, 1940 through December 31, 2039. If a 4-digit year is specified, the date may be in the range of August 24, 1928 through May 9, 2071. When it is passed to the CPP, it is always passed in the format *Cyyymmdd*, where C = century, yy = year, mm = month, and dd = day. The century digit is set to 0 for years 19xx, and it is set to 1 (one) for years 20xx. When a date value is specified in this PARM statement, it must be specified in one of the following formats: *mmdyy*, *mmdyyyy*, or *Cyyymmdd*. When a user enters a date in the command at run time, it must be entered in the job date format. The job date separator specifies the optional separator character that can be used when the date is entered. If the separator character is used, the date must be enclosed in apostrophes.

***TIME:** The parameter value is a character string that represents a time. It is entered in the format *hhmmss*. It is passed to the CPP in a 6-byte character string as *hhmmss*. The job time separator specifies the optional separator character that can be used when the time is entered. If the separator character is used, the time must be enclosed in apostrophes.

***CHAR:** The parameter value is a character string that (optionally) can be enclosed in apostrophes. If the character string contains any special characters (not including an asterisk (*)), it must be enclosed in apostrophes. The maximum length of the character string is 5000 characters.

***DEC:** The parameter value is a packed decimal number.

***LGL:** The parameter value is a logical value, either a one ('1') or a zero ('0').

***HEX:** The parameter value is in hexadecimal form. The specified characters must be 0 through F. They are converted to hexadecimal (EBCDIC) characters (2 hex digits per byte), right-justified, and padded on the left with zeros. If the value is enclosed in apostrophes, an even number of digits is required. If the value is not enclosed in apostrophes, an even number of digits is *not* required.

***ZEROELEM:** The parameter is always considered as a list of zero elements, for which no value can be specified in the command. It is used to prevent a value from being entered for a parameter that is a list even though the CPP expects one. For example, if two commands use the same CPP, one command could pass a list for a parameter and the other command may not have any values to pass. The second command would be coded with TYPE(*ZEROELEM).

***X:** (For IBM-supplied commands) The parameter value is a character string, variable name, or numeric value. The value is passed as a numeric value if it contains only digits, a "+" or "-" sign, and/or a decimal point; otherwise, it is passed as a character string.

***INT2:** The parameter value is an integer that is passed as a 2-byte signed binary number. CL programs do not support binary values in variables.

***INT4:** The parameter value is an integer that is passed as a 2-byte unsigned binary number. CL programs do not support binary values in variables.

***UINT2:** The parameter value is an integer that is passed as a 2-byte signed binary number. CL programs do not support binary values in variables.

***UINT4:** The parameter value is an integer that is passed as a 4-byte unsigned binary number. CL programs do not support binary values in variables.

***VARNAME:** (For IBM-supplied commands) The parameter value is a CL variable name that is passed as a character string. The name can contain a maximum of 11 characters, including the ampersand (&).

***CMD:** (For IBM-supplied commands) The parameter value is a command. For example, the IF command has a parameter called THEN whose value must be another command. The command is checked for validity by the command analyzer.

***NULL:** The parameter value is to be a null pointer, which can be used as a constant place holder. A DEP statement or the REL and RANGE keywords of other PARM statements may not reference the value of a parameter defined with TYPE(*NULL).

statement-label: Specify a qualified name or a mixed list of values. The statement label specified here by the TYPE parameter is the statement label that identifies the first of a series of QUAL or ELEM statements that further describe the qualified name or the mixed list being defined by this PARM statement.

LEN Parameter

Specifies the length of the parameter value that is passed to the CPP. LEN is not allowed if TYPE was specified as *INT2, *INT4, *DATE, *TIME, *CMD, *ZEROELEM, *NULL, or statement-label. With other TYPE specifications, this parameter has the following applications:

- If TYPE(*DEC) was specified, the decimal length is specified in the form (length1 length2), where length1 specifies the total number of digits in the value (including the decimal portion), and length2 specifies the number of allowable decimal digits to the right of the decimal point. (The value for length2 is optional. Zero is assumed if it is not entered.)

- If TYPE(*CHAR), TYPE(*NAME), TYPE(*SNAME), TYPE(*CNAME), TYPE(*CMDSTR), or TYPE(*VARNAME) was specified, only length1 is specified. It identifies the number of characters passed.
- If TYPE(*HEX) was specified, only length1 is specified. This length specifies the number of characters passed after the hexadecimal digits have been converted to character digits. Because 2 hexadecimal digits are converted to 1 decimal digit, the number of hexadecimal digits converted is twice the value of this length.
- If TYPE(*X) was specified, the LEN parameter is used as follows:
 - For character data, length1 specifies the minimum length to be passed. If a longer value is entered, the entire value is passed.
 - For decimal data, length2 and length3 specify the length and decimal positions for a constant value. If a variable is entered, it is passed according to the variable attributes.
 - For a logical value, length1 specifies the length of the value, which is always 1.

The default length that is assumed by the system and the maximum length for each type of parameter value are shown in the following table:

Data Type	Default Length	Maximum Length ¹
*DEC	(15 5)	(24 9)
*LGL	1	1
*CHAR	32	5000
*NAME	10	256
*SNAME	10	256
*CNAME	10	256
*PNAME	32	5000
*GENERIC	10	256
*HEX	1	256
*X	(1 15 5)	(256 24 9)
*VARNAME	11	11
*CMDSTR	256	20,000

Note:

¹ The maximum length shown here is the maximum length allowed by the Command Definition. The high-level language used as the CPP for the command may have different maximum lengths for these data types (for example, *DEC values in CL programs have a maximum length of (15 9)).

Whereas the maximum shown here is the maximum for the values used in the command when it is run, the maximum length of character constants specified in the command definition is limited to 32 characters. This restriction applies to the following parameters: CONSTANT, DFT, VALUES, REL, RANGE, SPCVAL, and SNGVAL.

For data types for which length cannot be coded, the following are the maximum lengths and the lengths passed.

Data Type	Maximum Length	Length Passed
*DATE ¹	8	7
*TIME ²	8	6
*ZEROELEM	0	0
*INT2 ³	6	2
*INT4 ⁴	11	4
*UINT2 ⁵	6	2
*UINT4 ⁶	11	4
*CMD	Any	Length needed
statement-label ⁵	–	–

Data Type	Maximum Length	Length Passed
Note:		
1	If a date is specified, the value is passed as 7 characters.	
2	If a time is specified, the value is passed as 6 characters.	
3	The value is passed as a 2-byte signed binary number and must meet the following condition: $-2^{15} < \text{or} = \text{value} < \text{or} = 2^{15}-1$	
4	The value is passed as a 4-byte signed binary number and must meet the following condition: $-2^{31} < \text{or} = \text{value} < \text{or} = 2^{31}-1$	
5	The value is passed as a 2-byte unsigned binary number and must meet the following condition: $0 < \text{or} = \text{value} < \text{or} = 2^{16}-1$	
6	The value is passed as a 4-byte signed binary number and must meet the following condition: $0 < \text{or} = \text{value} < \text{or} = 2^{32}-1$	
7	The length of the data accepted and passed is defined by the ELEM or QUAL statement that the label identifies.	

RTNVAL Parameter

Specifies whether a value is returned by the CPP through the parameter being defined in this PARM statement.

***NO:** No value can be returned in the parameter being defined. The parameter is an input parameter only.

***YES:** A value is to be returned by the CPP in the parameter. A CL variable name must be specified for the parameter to receive the value (when the command is called). If a variable name is not specified when the command is invoked, a null pointer is passed to the CPP.

- *YES is valid only if the TYPE parameter was specified as *DEC, *CHAR, *LGL, or *X. Also, *YES is valid only on commands that are limited to CL programs.
- *YES is not valid with MAX(>1), CONSTANT, DFT, RSTD, VALUES, REL, RANGE, SPCVAL, SNGVAL, FILE, FULL, EXPR, or ALWVAR(*NO).
- RTNVAL(*YES) can be specified if *BPGM, *IPGM, *BREXX, or *IREXX is specified in the CRTCMD command that uses the source file containing this PARM statement.
- VARY(*YES) must be specified if PASSATR(*YES) and RTNVAL(*YES) have been specified.

CONSTANT Parameter

Specifies that a value is passed to the CPP as a constant when the command being defined is processed. The parameter does not appear externally on the command. The value specified in this parameter (if any) must satisfy the requirements specified by the TYPE, LEN, VALUES, REL, RANGE, SPCVAL, and FULL parameters. As noted in the LEN parameter chart, if a character constant is specified in this parameter, it can be no longer than 32 characters.

If a constant is specified for the parameter being defined, no prompt text can be specified for the PROMPT parameter in the PARM statement, because the parameter will not be prompted.

The CONSTANT parameter is not valid with TYPE(*CMD), TYPE(*NULL), TYPE(*ZEROELEM), MAX(>1), DFT, RTNVAL(*YES), or EXPR(*YES).

Variables cannot be coded for this parameter.

RSTD Parameter

Specifies whether the value entered for the parameter (specified in the PARM statement) is restricted to only one of the values given in the VALUES, SPCVAL, or SNGVAL

parameters, or whether the value can be any value that satisfies the requirements specified by the TYPE, LEN, REL, RANGE, SPCVAL, SNGVAL, and FULL parameters.

***NO:** The value entered for the parameter specified by KWD in this PARM statement can be anything that matches the requirement specified by parameters TYPE, LEN, REL, RANGE, SPCVAL, SNGVAL and FULL in this PARM statement.

***YES:** The value entered for the parameter specified by KWD in this PARM statement is restricted to one of the values in the VALUES parameter, or to one of the from-values in the SPCVAL or SNGVAL parameters. *YES cannot be specified if TYPE(statement-label), TYPE(*CMD), TYPE(*NULL), TYPE(*ZEROELEM), TYPE(*X), or RTNVAL(*YES) is specified.

DFT Parameter

Specifies the default value that is assigned to the parameter if a value is not specified by the user. That is, the default value is used as the value of the parameter if the user omits the parameter while entering the command or if *N is entered as the parameter value. The default value must satisfy one of the following:

- It must match the requirements specified by the TYPE, LEN, REL, RANGE, and FULL parameters.
- It must be one of the from-values in the SPCVAL or SNGVAL parameters.
- If the default is a character constant, it can have no more than 32 characters (as noted in the LEN parameter chart).
- If RSTD(*YES) is specified, it must be in the list of values in the VALUES parameter or in the list of from-values of the SPCVAL or SNGVAL parameters.
- It must be a from-value in the SNGVAL parameter if the parameter being defined is a list of unlike values or it is a qualified name. This is true when a statement label is specified for TYPE; the label is used to identify a QUAL or ELEM statement.

The DFT parameter is not valid if CONSTANT is specified. The DFT parameter is valid only if MIN is 0, which means the parameter named in the KWD parameter is optional. A default cannot be specified if RTNVAL(*YES) is specified; instead, a null pointer is passed for the default. A default cannot be specified if TYPE(*CMD), TYPE(*ZEROELEM), or TYPE(*NULL) is specified. If TYPE(*VARNAME) is specified, a default special value can be specified; a default variable name cannot be specified.

If DFT is not specified and MIN(0) and RTNVAL(*NO) are specified, then the default assumed is as follows, depending on the specified type:

Assumed Default

	Parameter Types
0	*DEC *INT2 *INT4 *ZEROELEM
'0'	*LGL
zeros	*DATE *TIME *HEX
blanks	*CHAR *NAME *SNAME *CNAME *PNAME *GENERIC *VARNAME *CMDSTR
null	*CMD *X *NULL

An *assumed* default value is not displayed by the command prompt; a blank input field is shown instead. If a default is specified in the DFT parameter, it is displayed by the prompt exactly as specified.

value: Specify the default value that meets the specified requirements or that is one of the values specified in the VALUES, SPCVAL, or SNGVAL parameters.

Variables cannot be coded for this value.

VALUES Parameter

Specifies a list of up to 300 constants (fixed values) from which one constant can be entered as the value of the parameter named in the KWD parameter. The VALUES parameter is valid only if all of the following are true: RSTD(*YES) is specified, both the RANGE and REL parameters are not specified, and each constant matches the attributes specified by the TYPE, LEN, and FULL parameters. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. Enter the constants (not more than 300) that can be specified as the value of the parameter. The VALUES parameter is not valid if TYPE(*CMD), TYPE(*CMDSTR), TYPE(*X), TYPE(*NULL), TYPE(statement-label), TYPE(*VARNAME), or TYPE(*ZEROELEM), or if RTNVAL(*YES) is specified.

REL Parameter

Specifies the relationship between the parameter value of this parameter and the value of a constant or another parameter. If a keyword is specified, it must be preceded by an ampersand (&) to indicate that it is the value of the keyword that is to be tested. The value associated with the referenced keyword, not the user-specified value, is the value passed to the CPP. If the relationship is with another parameter whose value is a list of values or a qualified name, only the first value is used in the comparison.

To specify the relationship, enter one of the following relational operators followed by either a constant or the keyword name (KWD) of the other parameter (which must be preceded by an &).

- *LT less than
- *LE less than or equal to
- *EQ equal to
- *GE greater than or equal to
- *GT greater than
- *NL not less than
- *NE not equal to
- *NG not greater than

The REL parameter is not valid if RTNVAL(*YES) is specified, if either RANGE or VALUES is specified, or if TYPE is specified as *LGL, *VARNAME, *CMD, *CMDSTR, *X, *ZEROELEM, *NULL, or a statement label.

If a character type is specified by TYPE(*CHAR), the EBCDIC value of the character string is used as an unsigned integer in the comparison. As noted in the LEN parameter chart, if a character constant is specified in this parameter, it can be no longer than 32 characters.

Variables can be coded for this element.

RANGE Parameter

Specifies the range (the limits) for the value of the parameter. The parameter value must be greater than or equal to the lower limit value specified, and it must be less than or equal to the upper limit value specified. For example, 15 would be valid if RANGE was specified as (0 16).

For nonnumeric data types, such as *CHAR, the range of values and the data specified are left-justified and padded on the right with blanks. A numeric range should not be used to define an interval for nonnumeric data unless leading zeros are specified or the data is only 1 character long.

Variables can be coded for this element.

The upper and lower limits of the range can be specified either by a keyword representing the value or by the value itself. If a keyword is specified, it must be preceded by an ampersand (&) to indicate that the value of the keyword is to be tested. The value of its parameter at the time of the check is used to determine the range. The value that is tested is the value passed to the CPP, not the user-specified value. If the keyword identifies a list of values or a qualified name, only the first value is used as the range limit. A keyword may not reference a parameter that is defined with PASSVAL(*NULL), and RANGE is not valid with PASSVAL(*NULL).

The RANGE parameter is not valid if RTNVAL(*YES) is specified, if either REL or VALUES is specified, or if TYPE is specified as *LGL, *VARNAME, *CMD, *CMDSTR, *X, *ZEROELEM, *NULL, or statement label. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters.

Variables can be coded for this element.

SPCVAL Parameter

Specifies a list of up to 300 entries that define special values that can be entered on the parameter named in the KWD parameter. Each entry specifies a character string (a from-value) that can be entered even though it may not meet all validity checking requirements. If the entered character string matches the from-value of one of the entries, and the to-value is specified, the string is replaced with the to-value and is then passed to the CPP without further checking. If the to-value is omitted, the from-value is passed to the CPP. SPCVAL is not valid if TYPE is specified as *CMD, *CMDSTR, *X, *NULL, statement-label, or *ZEROELEM, or if RTNVAL(*YES) is specified.

The from-value is a character string, but the to-value can be anything that is passable. However, for TYPE(*DATE), the to-value must be specified unquoted in *mmdyy*, *mmdyyy*, or *Cyyymmdd* format. If a CL variable is used for the from-value, its type must be *CHAR. The to-value must be no longer than LEN specifies and, if TYPE is *DEC, *INT2, or *INT4, the type of the to-value must be the same; if TYPE is a character type (such as *CHAR, *LGL, or *DATE), the to-value must be a character string. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. If a to-value is not specified, the from-value must be passable.

If a to-value of *CURLIB is specified, the name of the current job library is passed to the CPP instead of the value *CURLIB. If the from-value is *CURLIB and no to-value is specified, or if the to-value is *CURLIB and it is enclosed in apostrophes, the value *CURLIB is passed to the CPP.

Variables cannot be coded for this element.

SNGVAL Parameter

Specifies a list of up to 300 single values that can be specified for a parameter being defined as a mixed list or as a qualified name (when a statement label is specified for TYPE), or specifies that it is to accept two or more values as defined by the MAX parameter. Any one of the single values can be used instead of a list of values or a qualified name that the parameter is defined to accept. Each entry specifies a character string (a from-value) that can be entered. If an entered character string matches the from-value of one of the entries and the to-value is specified, the data is replaced with the to-value and is passed to the CPP without further checking. If the to-value is omitted, the from-value is passed to the CPP.

The to-value (or the from-value, if the to-value is omitted) must be passable, as specified in the SPCVAL parameter. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. SNGVAL can be specified only if the MAX parameter is greater than 1 or TYPE is specified as a statement label of a QUAL or ELEM statement. Each single value can only substitute for a list of values or a qualified name; it cannot be a list element or qualifier. It is passed as the first and only element of the list.

SNGVAL is not valid if RTNVAL(*YES) is specified or if TYPE is specified as *CMD, *CMDSTR, *NULL, *ZEROELEM, or *X.

If a to-value of *CURLIB is specified, the name of the current job library, instead of the value *CURLIB, is passed to the CPP. If the from-value is *CURLIB and no to-value is specified, or if the to-value is *CURLIB and it is enclosed in apostrophes, the value *CURLIB is passed to the CPP.

Variables cannot be coded for this element.

MIN Parameter

Specifies the minimum number of values that must be entered for the parameter being defined.

For a parameter that does not allow multiple like values, only zero (0) for optional and 1 for required can be specified as the minimum number of values.

Note: Required parameter statements must precede optional statements. If required parameter statements are not specified first, the system assumes that the specified parameter is optional, and the minimum number of values for required parameters is ignored.

For a parameter that allows multiple like values (because a value greater than 1 is specified for the MAX parameter), zero (0) indicates that no values need be entered; therefore, it is an *optional* parameter. A value equal to or greater than one (1) indicates the minimum number of values that must be entered for the parameter; therefore, it is a *required* parameter. The value exceeds the value specified for the MAX parameter and cannot exceed 1 for TYPE(*NULL).

0: The parameter is optional; it does not have to be entered.

minimum-number: Specify the minimum number of elements that must be specified for this parameter (named by the KWD parameter). If 1 is the assigned value, it specifies that at least one value is required for the parameter. If a number greater than 1 is specified, the parameter is a list that must have at least as many elements as the number specified.

MAX Parameter

Specifies, if this PARM statement is defining a simple list parameter, the maximum number of list elements that this list parameter can contain. If a value greater than 1 is specified, the parameter is capable of accepting multiple like values (that is, a simple list). This support is primarily intended for IBM-supplied commands. All values entered for this parameter (at the time the command is run) must satisfy the validity checking requirements specified by the other parameter values on this PARM statement.

Note: The values for a list parameter are passed consecutively, preceded by a 2-byte binary value that indicates the number of values entered in the parameter by the user. CL programs do not support the handling of binary values in variables.

1: The parameter accepts only one value; the parameter is not a list parameter.

maximum-number: Specify the maximum number of elements that the list parameter can accept. The specified maximum must be greater than or equal to the value specified in MIN, and less than or equal to 300. If the maximum is greater than 1 and TYPE is not a statement label that identifies a QUAL or ELEM statement, the parameter is a simple list of

like elements (that is, each element in the list has the same requirements, such as type and length). If TYPE(statement-label) is specified and it points to the label of an ELEM or QUAL statement, MAX should only be specified greater than 1 if a list of lists or a list of qualified names is accepted. A maximum greater than 1 is not valid if TYPE is specified as *CMD, *CMDSTR, or *NULL, or if RTNVAL(*YES) or CONSTANT is specified.

ALWUNPRT Parameter

Specifies whether this PARM statement accepts the hexadecimal characters X'FF' or those in the range of X'00' to X'3F'. This parameter is valid only for TYPE(*CHAR) or TYPE(*X).

***YES:** Any characters can be passed to the CPP and sent to the display or printer.

***NO:** Unprintable characters cannot be passed to the CPP.

ALWVAR Parameter

Specifies whether to allow variable names for the parameter. If TYPE was specified as *VARNAME, *ZEROELEM, *NULL, or statement-label, ALWVAR(*NO) is not allowed.

***YES:** Variable names can be used for the parameter.

***NO:** Variable names cannot be used for the parameter.

PGM Parameter

Specifies whether this parameter element is a program name. PGM(*YES) is valid only for statement-label, *CHAR, *NAME, *SNAME, *CNAME, and *GENERIC types. The specification of the PGM(*YES) parameter does not have any effect on the parameter element being defined by the PARM statement; it only indicates to the compiler that the value for this parameter is a program name. This information is stored so it can be included in the output of the Display Program References (DSPPGMREF) command.

***NO:** The parameter defined in this PARM statement is not a program name.

***YES:** The parameter defined in this PARM statement is a program name.

DTAARA Parameter

Specifies whether the parameter is a data area name. DTAARA(*YES) is valid only for statement-label, *CHAR, *NAME, *SNAME, *CNAME, and *GENERIC types. The specification of the DTAARA(*YES) parameter does not have any effect on the parameter being defined by the PARM statement; it only indicates to the compiler that the value for this parameter is a data area. This information is stored so it can be included in the output of the Display Program References (DSPPGMREF) command.

***NO:** The parameter defined in this PARM statement is not a data area name.

***YES:** The parameter defined in this PARM statement is a data area name.

FILE Parameter

Specifies whether the parameter is a file name and the expected use of the file. The parameter can be specified as the name of a file that has a specific use so that, at compile time, the names can be used to get file reference information about where the files are used. The specification in the FILE parameter does not have any effect on the operation of the parameter being defined; it only indicates to the compiler that the value for this parameter is a file name and what type of file it is. This information is stored so it can be included in the output of the Display Program References (DSPPGMREF) command. The FILE parameter is valid only if *CHAR, *NAME, *SNAME, *CNAME, *GENERIC, or statement-label is specified for the TYPE parameter. The FILE parameter is not valid with RTNVAL(*YES). One of the following types of files can be specified:

***NO:** The parameter (named by KWD) is not a file name.

***IN:** The parameter value is an input file name.

***OUT:** The parameter value is an output file name.

***UPD:** The parameter value is an update file name.

***INOUT:** The parameter value is the name of a file that is used for both input and output.

***UNSPFD:** The parameter value is the name of a file, but its use cannot be specified.

The use of the file must match the type of file specified. For example, if *IN is specified, the file can be used only for input; if *UPD is specified, it can be used only to update existing records.

FULL Parameter

Specifies whether the number of characters in the parameter must be exactly the same as the number specified in the LEN parameter (if specified) or its default length (if LEN is not specified).

***NO:** The number of characters in the parameter can be less than that specified by the LEN parameter.

***YES:** The number of characters in the parameter must equal the number specified by LEN or the default length for that type. The exact length is valid only for the following parameter types: *LGL, *CHAR, *NAME, *SNAME, *CNAME, *GENERIC, *VARNAME, and *HEX. FULL(*YES) is valid with RTNVAL(*YES).

EXPR Parameter

Specifies whether the parameter named in the KWD parameter can accept an expression containing a character concatenation or a built-in function (%SUBSTRING or %BIN). Valid character concatenation operators are as follows:

Concatenation	*CAT or,
Blank insertion with concatenation	*BCAT or, >
Blank truncation with concatenation	*TCAT or, <

Restrictions: Expressions are not allowed on parameters where the TYPE parameter specifies *CMDSTR, *CMD, *ZEROELEM, *NULL, or statement-label.

***NO:** The parameter value cannot be a concatenation expression or a built-in function.

***YES:** The parameter value can be a concatenation expression or a built-in function.

VARY Parameter

Specifies whether the value that is passed to the CPP is preceded by a length value that indicates the number of characters entered for the command parameter.

The length value is the actual number of characters entered for the command parameters with trailing blanks removed. The length value passed may be different than the defined parameter length or the declared variable length. The length of the field containing the character string data is determined by the defined length for the parameter or the declared LEN for CL program variables. The length value defines how many characters in the character string data field were actually entered for the command parameter. This parameter may be specified as a single value (*NO) or as a list of two values (elements).

***NO:** The parameter value is not preceded by a length value.

Element 1: Return Length Value:

***YES:** The parameter value passed to the CPP is preceded by a field that indicates the number of characters actually specified for the parameter.

Note: *YES is valid only for the following parameter types: *CHAR, *NAME, *SNAME, *CNAME, *PNAME, *GENERIC, *LGL, *VARNAME, *CMD, *CMDSTR, and *X. *YES must be specified if PASSATR(*YES) and RTNVAL(*YES) are specified.

Element 2: Value Length:

***INT2:** The length of the parameter value is an integer passed as a 2-byte signed binary number.

***INT4:** The length of the parameter value is an integer passed as a 4-byte signed binary number.

PASSATR Parameter

(For IBM-supplied commands) Specifies whether an attribute byte is to be passed to the CPP with the parameter data.

***NO:** No attribute byte is passed with the parameter.

***YES:** An attribute byte is passed with the parameter; the attribute byte indicates whether the data value came from the default, the data type of the value, and, if TYPE(*CHAR) was specified, whether the character string was enclosed in apostrophes.

PASSVAL Parameter

Specifies whether a value is to be passed to the command processing program for this parameter. *NULL is not valid if the parameter is a constant parameter (a parameter in which a value has been specified for the CONSTANT parameter on the PARM statement, a parameter for which TYPE(*ZEROELEM) or TYPE(*NULL) has been specified, or is a list/qualified name defined by all constant ELEM or QUAL statements). *NULL also is not valid if RTNVAL(*YES) has been specified or if the value specified for the MIN parameter is greater than zero. A DEP statement or the REL and RANGE keywords of other PARM statements may not reference the value of a parameter defined with PASSVAL(*NULL).

***DFT:** The default value is always passed to the CPP.

***NULL:** A null pointer is passed to the CPP if the parameter is not specified.

CASE Parameter

Specifies whether the value that is passed to the CPP is changed from lowercase to uppercase, or is preserved in the case specified on the command parameter.

***MONO:** The parameter value is changed from lowercase to uppercase. Parameters with apostrophes preserve the case whether or not this value is specified.

***MIXED:** The parameter value is preserved in the case specified on the command parameter. This value can be specified only on *CHAR and *PNAME parameter types.

LISTDSPL Parameter

Specifies whether the displacement to a list within a list is 2-bytes or 4-bytes long. These displacements are generated when a parameter being passed to a CPP has a list within a list. This parameter is ignored if the value being built for the CPP does not contain a list within a list.

***INT2:** The displacement value is an integer passed as a 2-byte signed binary number.

***INT4:** The displacement value is an integer passed as a 4-byte signed binary number.

DSPINPUT Parameter

Specifies whether the keyword value is shown in the job log or on a prompt display. DSPINPUT(*PROMPT) and DSPINPUT(*NO) are valid for the following parameter types: *CHAR, *NAME, *SNAME, *CNAME, *GENERIC, *DEC, *LGL, *INT2, *INT4, *DATE, *TIME, *HEX, and *CMDSTR.

Note: The DSPINPUT parameter has no effect on the job log entries for a database reader job, for imbedded commands (for example, a command submitted on the SBMJOB command), or for commands executed using the QCMDEXC or QCAEXEC APIs.

***YES:** The parameter value is displayed on the prompt display and in the job log.

***PROMPT:** The parameter value is displayed on the prompt display but not in the job log.

***NO:** The parameter value is shown in neither the job log or a prompt display. When a previously entered command is retrieved, the nondisplay field entries must be retyped (their previous values are not retrievable). When a job log entry is created, the nondisplay field is replaced by empty parentheses ().

CHOICE Parameter

Specifies the text that is displayed to the right of the prompt line of each parameter on the prompt screen. Up to 30 characters of text can be displayed.

***VALUES:** Each possible value is displayed in the possible values field, separated by a comma and a space. If values are specified for the Default, Single value, or Special value parameters, the first value displayed is the default value; the next value is a single value, and the values following that are special values. If there are too many values to fit in 30 characters, the last value is followed by three periods.

Examples of possible values text follow:

- If *NO is specified on the RSTD parameter and *DEC is specified on the TYPE parameter and the RANGE parameter is not specified, the word "RANGE" is displayed in the possible values field. The resulting line will appear in the form: RANGE, *XXX, *YYY, *ZZZ...
- If *NO is specified on the RSTD parameter and *DEC is specified on the TYPE parameter and the RANGE parameter is specified, the range of possible values is displayed in the possible values field. The resulting line will appear in the form: a-b, *XXX, *YYY, *ZZZ (where a and b are numerals defining the range).
- If *YES is specified on the RSTD parameter, the possible values displayed are determined by the VALUES parameter, the SNGVAL parameter, and the SPCVAL parameter. The resulting line will appear in the form: *XXX, *YYY, *ZZZ...

***NONE:** No values are displayed.

***PGM:** A program that is called determines the values that are displayed. The program that is called is identified in CHOICEPGM parameter.

message-identifier: Specify the message ID of the message used to retrieve the message containing the text for the possible values field. The message file specified on the PMTFILE parameter of the Create Command (CRTCMD) command is used to find the message.

'choices-text': Specify no more than 30 characters, enclosed in apostrophes.

CHOICEPGM Parameter

Specifies the qualified name of the program that is called during the prompting to fill in the possible choices text and the permissible values during prompting. This parameter must be specified if CHOICE(*PGM) is specified, and it may not be specified otherwise.

***NONE:** No program is identified to fill in the possible choices text and permissible values.

The possible library values are:

***LIBL:** The library list is used to locate the program name.

***CURLIB:** The current library for the job is used to locate the program name. 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 where the program name is located.

program-name: Specify the name of the program called during prompting to fill in the possible choices text or a permissible value.

If an exception occurs when the program is called, no possible choices text will be left blank, and the list of permissible values will be taken from the command.

PMTCTL Parameter

Specifies how prompting is controlled for this parameter. Prompting may be conditioned by another parameter, specified by a PMTCTL statement referred to by label in this parameter, or called for by pressing the F10 key.

***NONE**: Specifies that this parameter is always prompted, unless it is omitted due to selective prompting.

***PMTRQS**: Specifies that this parameter is not prompted unless:

- The user requests additional parameters.
- A value was entered for the parameter before the prompt was called.

statement-label: Specifies the label of the PMTCTL statement that is used to determine whether this parameter is prompted. The parameter is not prompted unless:

- The conditions specified on the referred to PMTCTL statement have been met.
- A value was entered for the parameter before the prompt was called.

PMTCLPGM Parameter

Specifies the qualified name of the program called to convert the value specified for the parameter into a value used on a PMTCTL statement. This parameter is valid only on parameters that are referred to in the CTL parameter of a PMTCTL statement.

***NONE**: There is no program to convert the parameter value for prompt control statements. If the parameter is specified in a PMTCTL statement, the actual value is compared in the PMTCTL statement.

The possible library values are:

***LIBL**: The library list is used to locate the program name.

***CURLIB**: The current library for the job is used to locate the program name. 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 where the program name is located.

program-name: Specifies the qualified name of the program called to convert the parameter value.

KEYPARM Parameter

Specifies whether the parameter is a key parameter. Key parameters can be used only if a prompt override program is specified on the PMTOVRPGM parameter of the CRTCMD or CHGCMD commands. The prompt override program overrides the command processing program (CPP) by showing only the key parameters on the initial prompt display. Values must be input for these parameters before the remaining parameters are shown. The remaining parameters are shown on the prompt display with the actual values, instead of *SAME or *PRV.

***NO**: The parameter is not a key parameter.

***YES:** The parameter is a key parameter. The following rules are followed if *YES is specified:

1. Key parameters must be placed before non-key parameters in the command definition statement.
2. Key parameters appear on the prompt display in the same order as they appear in the command definition statement.
3. Key parameters are valid only if a prompt override program is specified on the PMTOVRPGM parameter of the CRTCMD or CHGCMD commands; if a prompt override program is not specified, the key parameters are treated as non-key parameters, and a warning message is issued.
4. Key parameters must not be the only parameters in the command definition statement; if they are, a warning message is sent.

INLPMTLEN Parameter

Specifies the length of the input field initially displayed for the parameter when the command is prompted. The user can extend the field to a maximum length of 512 bytes by entering an ampersand (&) in the first position of the field, followed by a blank. INLPMTLEN is valid only if TYPE is specified as *CHAR, *NAME, *SNAME, *CNAME, *PNAME, *GENERIC, *CMDSTR, *HEX, *X, or *CMD. If FULL(*YES), RTNVAL(*YES), RSTD(*YES), or CONSTANT are specified, INLPMTLEN(*CALC) must be specified or defaulted.

***CALC:** The prompter will determine the length of the prompt field based on the type and length of the parameter.

***PWD:** If the current value of system value QPWDLV is '0' or '1', the prompt field will be 10 bytes long. Otherwise, the length of the prompt field will be determined by the length of the parameter. INLPMTLEN(*PWD) is valid only if type is specified as *CHAR, *NAME, *SNAME, *PNAME, or *CNAME.

initial-prompt-length: Specify the initial length in bytes of the prompt field. Valid values are 1-12, 17, 25, 32, 50, 80, 132, 256, and 512.

PROMPT Parameter

Specifies the prompt text, if any, that is used for the parameter named by the KWD parameter. The prompt text further describes the keyword and input field to the user, who may enter a response to the information displayed. For example, the prompt text for the TYPE parameter on the Create Library (CRTLIB) command is:

Library type

When the prompt for the command is displayed, the prompt for the TYPE parameter is:

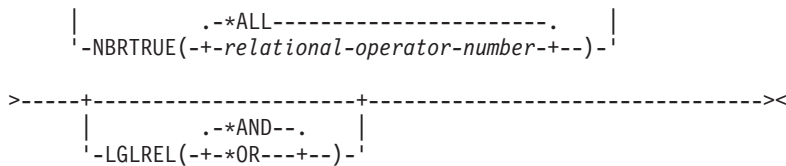
Library type *PROD

The underscored field is the prompt input field where the user enters the value for the TYPE parameter. When the prompt is displayed, the prompt input field contains the default value (*PROD in this example). Prompt text cannot be specified if TYPE(*ZEROELEM) or TYPE(*NULL) is specified or if a constant value is specified in the CONSTANT parameter.

***NONE:** No prompt text is displayed for the parameter defined by this PARM statement. This parameter is still asked for by its keyword name, but no text is displayed with it.

message-identifier relative-prompt-number: Specify the message identifier that specifies the message that contains the prompt text of up to 30 characters that is displayed when the parameter is prompted. If a message having the specified identifier cannot be found in the message file specified in the PMTFILE parameter of the Create Command (CRTCMD) command, the message identifier itself is used as the prompt text.

Optionally, a relative prompt number can be specified with the message identifier. The relative prompt number specifies the order in which parameter keywords are to be asked



Note:

1. A maximum of 50 repetitions

CTL Parameter

Specifies the name of the parameter that controls the prompting. The value of the parameter specified here is compared to the value specified in the COND parameter. If the PMTCTLPGM parameter is coded for the parameter specified here, the value returned by the program specified in the PMTCTLPGM parameter is compared to the values specified in the COND parameter. If the parameter specified here is a list or qualified name, only the first list element or qualifier is compared.

COND Parameter

Specifies the condition against which the parameter specified in the CTL parameter is tested. Up to 50 conditions can be specified.

***SPCFD:** The condition is true, including the default value, if it is specified for the control parameter.

***UNSPCFD:** The condition is true only if the control parameter is not specified. It is not true if the default value is specified.

relational-operator value: Specify the relational operator and value used to compare the value of the control parameter to the value specified in the COND parameter. Relational operators are *GT, *EQ, *NL, *LT, *NE, *LE, and *NG.

NBRTRUE Parameter

Specifies the number of conditions (indicated in the COND parameter) that must be true if the parameter is prompted.

***ALL:** All the conditions must be true.

relational-operator number: Specify the relational operator and number used to compare the number of conditions that are true to the number specified in the NBRTRUE parameter. Relational operators are *GT, *EQ, *NL, *LT, *NE, *LE, and *NG.

LGLREL Parameter

Specifies, when multiple PMTCTL statements are in a group, the logical relationship of the statement to the previous statements in the group. This parameter is not allowed on the first PMTCTL statement in a group.

***AND:** This statement is in an AND relationship of the statement to the previous statements in the group.

***OR:** This statement is in an OR relationship with the previous PMTCTL statement or statements. Statements in an OR relationship are evaluated after AND relationships are evaluated.

Example: PMTCTL statement

```

A: PMTCTL CTL(TYPE) COND((*EQ *) (*EQ *LIST))
  NBRTRUE(*EQ 1)

```

If TYPE(*) or TYPE(*LIST) is specified, the parameters which reference this PMTCTL statement are selected for prompting.

```

B: PMTCTL  CTL(P1)  COND((*EQ *ALL))
    PMTCTL  CTL(P1)  COND((*EQ *SOME)) LGLREL(*OR)
    PMTCTL  CTL(P2)  COND((*EQ *ALL)) LGLREL(*AND)
    PMTCTL  CTL(P1)  COND((*EQ *NONE)) LGLREL(*OR)
    PMTCTL  CTL(P2)  COND((*NE *ALL)) LGLREL(*AND)

```

The parameters which refers to this group of PMTCTL statements are selected for prompting if any of the following conditions exist:

- *ALL is specified for P1.
- *SOME is specified for P1 and *ALL is specified for P2.
- *NONE is specified for P1 and *ALL is not specified for P2.

QUAL (Qualifier) statement

The Qualifier (QUAL) statement describes one part of a qualified name. If a name is the allowed value of a parameter or list element defined in a PARM or ELEM statement, it can be changed to a qualified name by using a QUAL statement for each qualifier used to qualify the name.

The order in which the QUAL statements are entered into the source file determines the positional order in which the qualifiers must be specified and passed to the validity checker and the command processing program (CPP).

The QUAL statement (or only the first QUAL statement if there is more than one) *must* have a statement label that matches the statement label value that must be specified in a PARM or ELEM statement for which the qualifier is being defined. The qualifiers for the parameter or list element are then entered on the command in the form value3/value2/value1 where values 1 through 3 are qualifiers described by a QUAL statement.

Note: In the System/38 environment, the values are passed in sequential order with periods as delimiters. An example of this form is value1.value2.value3.

The values are passed with each value padded to its defined length and with the delimiters (slashes or periods) removed.

Note: The QUAL statement contains certain parameters and predefined values that can be used only when IBM-supplied CPPs are called by the command being defined. Limitations in some high-level languages reduce the usefulness of these values in the definition statements of user-defined commands. These parameters and values are identified by the phrase “(For IBM-supplied commands)” that immediately follows the parameter keyword (if the entire parameter is for IBM-supplied commands only) or the predefined value to which it applies.

The following syntax diagram shows the syntax for the QUAL definition statement.

```

>>-QUAL---TYPE--(--+*NAME-----+---)-----+-----+-----+----->
      +-*SNAME---+          '-LEN--(--length--)--'
      +-*CNAME---+
      +-*GENERIC-+
      +-*CHAR----+
      +-*INT2----+
      +-*INT4----+
      +-*UINT2---+
      '-*UINT4---'

>-----+-----+-----+-----+-----+-----+-----+----->
      '-CONSTANT--(--value--)--' |          .-*NO--. |
                                '-RSTD--(--+*YES-+---)--'

>-----+-----+-----+-----+-----+-----+-----+----->
      '-DFT(----value-----)--'

```


TYPE Parameter

Specifies the type of qualifier used to qualify a parameter name or list element name. The qualifier can be a name or generic name, a quoted or unquoted character string, or an integer. Enter one of the following options to specify the type of qualifier. The first qualifier for any qualified name must have a type of name (*NAME) or generic name (*GENERIC).

***NAME:** The qualifier is a character string that represents a name. The maximum length of the name is 256 characters. The first character must be alphabetic (including the special characters \$, #, or @), and the remaining characters must be alphanumeric, a period, or an underscore. The name must be a string of characters starting and ending with double quotation marks ("). If a special value is used (as in *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***SNAME:** The qualifier is a character string that represents a name. The maximum length of the name is 256 characters. The first character must be alphabetic (including the special characters \$, @, or #), and the remaining characters must be A through Z, 0 through 9, or \$, #, or @. Periods (.) are not allowed. If a special value is used (as in *LIBL or *NONE), it should be specified in the SPCVAL parameter.

***CNAME:** The qualifier is a character string that represents a name. The maximum length of the name is 256 characters. The first character must be alphabetic (including the special characters \$, #, or @), and the remaining characters must be A through Z, 0 through 9, \$, #, or @. Periods (.) and underscores (__) are not allowed. If a special value is used (as in *LIBL or *NONE), it must be specified in the SPCVAL parameter.

***GENERIC:** The qualifier is a character string that represents a generic name. A generic name contains a maximum of 255 characters followed by an asterisk (*) or 256 characters without an asterisk. The name identifies a group of objects whose names all begin with the characters preceding the *. If an asterisk (*) is not included, the system assumes that the generic name is a complete object name.

***CHAR:** The qualifier is a character string that can (optionally) be enclosed in apostrophes. If the character string contains any special characters (not including an asterisk (*)), it must be enclosed in apostrophes. The maximum length of the character string is 5000 characters.

***INT2:** The qualifier is an integer that is passed as a 2-byte signed binary number. CL programs do not support binary values in variables.

***INT4:** The qualifier is an integer that is passed as a 4-byte signed binary number. CL programs do not support binary values in variables.

LEN Parameter

Specifies the length of the qualifier, if its data type is *NAME, *GENERIC, or *CHAR. The default length that is assumed by the system and the maximum length for each type of qualifier are shown in the following table:

Data Type	Default Length	Maximum Length ¹
*NAME	10	256 ²
*SNAME	10	256 ²
*CNAME	10	256 ²
*GENERIC	10	256 ²
*CHAR	32	5000 ²
*INT2 ³	6	
*INT4 ⁴	11	

Data Type	Default Length	Maximum Length ¹
-----------	----------------	-----------------------------

- Note:
- ¹ The maximum length shown here is the maximum length allowed by Command Definition. The high-level language used as the CPP for the command may have a different maximum lengths for these data types (for example, *DEC values in CL programs have a maximum length of (15 9)).
 - ² Whereas the maximum shown here is the maximum for values used in the command when it is run, the maximum length of character constants specified in the command definition is limited to 32 characters. This restriction applies to the following parameters: CONSTANT, DFT, VALUES, REL, RANGE, and SPCVAL.
 - ³ For *INT2, length cannot be specified; its assumed length is 6. The value must meet the following condition: $-2^{15} \leq \text{value} \leq 2^{15}-1$. The value is passed as a 2-byte signed binary number.
 - ⁴ For *INT4, length cannot be specified; its assumed length is 11. The value must meet the following condition: $-2^{31} \leq \text{value} \leq 2^{31}-1$. The value is passed as a 4-byte signed binary number.

CONSTANT Parameter

Specifies that a value is passed to the CPP as a constant for the qualifier when the command being defined is processed; the qualifier does not appear externally on the command. If specified, the value must satisfy the requirements specified by the TYPE, LEN, VALUES, REL, RANGE, and SPCVAL parameters. (As noted in the LEN parameter chart, if a character constant is specified in this parameter, it can be no longer than 32 characters.)

If a constant is specified in this QUAL statement, and other QUAL statements immediately follow it, they must also be defined as constants, unless a label precedes one of them. A label indicates the beginning of a new group of QUAL statements, which can be defined differently.

Also, if a constant is specified for the qualifier being defined, no prompt text can be specified for the PROMPT parameter of this QUAL statement. However, any other qualifiers or groups of qualifiers are still prompted, and their values are still passed to the CPP as a qualified name.

The CONSTANT parameter is not valid if the DFT parameter is specified or if *YES is specified on the EXPR parameter.

Variables cannot be coded for this parameter.

RSTD Parameter

Specifies whether the value entered for the qualifier is restricted to only one of the values given in the VALUES or SPCVAL parameters, or whether any value can be used that satisfies the requirements specified by the TYPE, LEN, REL, RANGE, and SPCVAL parameters.

***NO:** The value entered for the qualifier defined by this QUAL statement can be anything that satisfies the requirements specified by the TYPE, LEN, REL, RANGE, and SPCVAL parameters in this QUAL statement.

***YES:** The value entered for the qualifier defined by this QUAL statement is restricted to one of the values in the VALUES parameter, or to one of the from-values in the SPCVAL parameters.

DFT Parameter

Specifies the default value assigned to the qualifier if a value is not specified by the user. The default value must satisfy one of the following:

- It must match the qualifier requirements specified by the TYPE, LEN, REL, and RANGE parameters.
- It must be one of the from-values in the SPCVAL parameter.

- If RSTD(*YES) is specified, it must be in the list of values in the VALUES parameter or in the list of from-values in the SPCVAL parameter.
- If the default is a character constant, it can have no more than 32 characters as noted in the LEN parameter chart.

The DFT parameter is valid only if the MIN parameter is 0, which means the qualifier defined by this QUAL statement for this list is optional. A default is not meaningful on this QUAL statement if it is the first one (defining the first part) for a qualified name and if a default is specified on the PARM or ELEM statement that this QUAL statement further defines.

If DFT is not specified, it has a default of its own: a blank () if TYPE was specified as *CHAR, *NAME, *SNAME, *CNAME, or *GENERIC; or a zero (0) if TYPE was specified as *INT2 or *INT4. An assumed default value is not displayed by the command prompt; a blank input field is shown instead. If a default is specified in the DFT parameter, it is displayed by the prompt exactly as specified.

The DFT parameter is not valid if the CONSTANT parameter is specified.

value: Specify the default value that meets the specified requirements or that is one of the values specified in the SPCVAL or VALUES parameters.

Variables cannot be coded for this value.

VALUES Parameter

Specifies a list of up to 300 constants (fixed values) from which one constant can be entered as the value of the qualifier. The VALUES parameter is valid only if all of the following are true: RSTD(*YES) is specified, both RANGE and REL are *not* specified, and the constant matches the attributes specified by the TYPE and LEN parameters in this QUAL statement. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. Enter the constants (not more than 300) that can be specified as the value of the qualifier.

REL Parameter

Specifies the relationship between the qualifier value and the value of another parameter or constant. To specify the relationship, enter one of the following relational operators followed by a constant or the value of another parameter.

- *LT less than
- *LE less than or equal to
- *EQ equal to
- *GE greater than or equal to
- *GT greater than
- *NL not less than
- *NE not equal to
- *NG not greater than

The REL parameter is not valid if either RANGE or VALUES is specified. If a character type is specified by TYPE(*CHAR), the EBCDIC value of the character string is used as an unsigned integer in the comparison. As noted in the LEN parameter chart, if a character constant is specified in this parameter, it can be no longer than 32 characters.

RANGE Parameter

Specifies the range (limits) for the value of the qualifier. The qualifier value must be

greater than or equal to the lower limit value specified, and it must be less than or equal to the upper limit value specified. For example, 15 would be valid if the RANGE parameter was specified as (0 16). For nonnumeric data types, such as *CHAR, the range of values and the data specified is left-justified and padded on the right with blanks. A numerical range should not be used to define an interval for nonnumeric data unless leading zeros are specified or the data is only 1 character in length. The RANGE parameter is not valid if either the REL or VALUES parameter is specified. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters.

SPCVL Parameter

Specifies a list of up to 300 entries that define special values that can be entered on the parameter named in the KWD parameter on the PARM statement. Each entry specifies a character string (a from-value) that can be entered even though it may not meet all validity checking requirements. If the entered character string matches the from-value of one of the entries, and the to-value is specified, the string is replaced with the to-value and is then passed to the CPP without further checking. If the to-value is omitted, the from-value is passed to the CPP. The from-value is a character string, but the to-value can be anything that is passable. If a CL variable is used for the from-value, its type must be *CHAR.

However, the first qualifier can only have special to-values with the from-values that are a name, a generic name, or an asterisk, followed by a name, such as *ALL.

Each to-value must be passable to the CPP. The to-value must be no longer than the LEN parameter specifies and, if TYPE is *INT2 or *INT4, the type of the to-value must be the same; if TYPE is a character type (such as *CHAR or *NAME), the to-value must be a character string. As noted in the LEN parameter chart, character constants specified in this parameter can be no longer than 32 characters. If a to-value is not specified, the from-value must be passable.

If a to-value of *CURLIB is specified, the name of the current job library is passed to the CPP instead of the value *CURLIB. If the from-value is *CURLIB and no to-value is specified, or if the to-value is *CURLIB and it is enclosed in apostrophes, the value *CURLIB is passed to the CPP.

Variables cannot be coded for this value.

ALWUNPRT Parameter

Specifies whether this QUAL statement should accept the hexadecimal characters X'FF' or those in the range of X'00' to X'3F'. This parameter is valid only for TYPE(*CHAR) or TYPE(*X).

***YES:** Any characters can be sent to the display or printer.

***NO:** Unprintable characters cannot be passed to the command processing program.

ALWVAR Parameter

Specifies whether to allow variable names for the qualifier. If TYPE was specified as *VARNAME, *ZERO, *NULL, or statement-label, ALWVAR(*NO) is not allowed.

***YES:** Variable names can be used for the qualifier.

***NO:** Variable names cannot be used for the qualifier.

MIN Parameter

Specifies whether the qualifier being defined in this QUAL statement is required or optional. If MIN is not specified, 0 is assumed, which means the qualifier is optional. If a required qualified name is needed, MIN(1) must be coded on both the first QUAL statement and on the PARM or ELEM statement that references it.

0: The qualifier is optional on the name being qualified.

1: The qualifier is required on the name being qualified; it must be entered.

FULL Parameter

Specifies whether the number of characters in the qualifier value must be exactly the same as the number specified in the LEN parameter (if specified) or its default length (if LEN is not specified).

***NO:** The number of characters in the qualifier value can be less than that specified by the LEN parameter.

***YES:** The number of characters in the qualifier value must equal the number specified by the LEN parameter or the default length for that type. The exact length is valid only for the *CHAR, *NAME, and *GENERIC qualifier types.

EXPR Parameter

Specifies whether the qualifier can accept an expression containing a character concatenation. Valid character concatenation operators are as follows:

Concatenation	*CAT or,
Blank insertion with concatenation	*BCAT or, >
Blank truncation with concatenation	*TCAT or, <

***NO:** The qualifier value cannot be a concatenation expression.

***YES:** The qualifier value can be a concatenation expression.

VARY Parameter

Specifies whether the qualifier value passed to the CPP is preceded by a length value that indicates the number of characters entered for the qualifier's value.

***NO:** The qualifier value is not preceded by a length value.

***YES:** The qualifier value passed to the CPP is preceded by a 2-byte binary length field that indicates the number of characters actually specified for the qualifier. The data is passed in a field of the length specified by LEN or by the default length. *YES is valid only for the following qualifier types: *CHAR, *NAME, *SNAME, *CNAME, or *GENERIC. If a CL variable is specified for this qualifier, the 2-byte binary length field contains the length of the variable value with trailing blanks removed, not the declared length of the CL variable.

PASSATR Parameter

(For IBM-supplied commands) Specifies whether an attribute byte is to be passed to the CPP with the qualifier.

***NO:** No attribute byte is passed with the qualifier.

***YES:** An attribute byte is passed with the qualifier; the attribute byte indicates whether the data value came from the default, the data type of the value, and, if TYPE(*CHAR) was specified, whether the character string was enclosed in apostrophes.

DSPINPUT Parameter

Specifies whether the keyword value is shown in the job log or on a prompt screen.

Note: The DSPINPUT parameter will have no effect on the job log entries for a database reader job, for imbedded commands (for example, a command submitted on the SBMJOB command), or for commands executed using the QCMDEXC or QCAEXEC APIs.

***YES:** The parameter value is shown on the prompt display and in the job log.

***PROMPT:** The parameter value is shown on the prompt display but not in the job log.

***NO:** The parameter value is not shown either in the job log or on a prompt display. When a previously entered command is retrieved, the nondisplay field entries must be retyped (their previous values are not retrievable). When a job log entry is created, the nondisplay field is replaced by empty parentheses ().

CHOICE Parameter

Specifies the text that is displayed to the right of the prompt line of each parameter on the prompt screen. Up to 30 characters of text can be displayed.

***VALUES:** Each possible value is displayed in the possible values field, separated by a comma and a space. If values are specified for the Default, Single value, or Special value parameters, the first value displayed is the default value; the next value is a single value, and the values following that are special values. If there are too many values to fit in 30 characters, the last value is followed by three periods.

Examples of possible values text follow:

- If *NO is specified on the RSTD parameter and *DEC is specified on the TYPE parameter and the RANGE parameter is not specified, the word "RANGE" is displayed in the possible values field. The resulting line will appear in the form: RANGE, *XXX, *YYY, *ZZZ...
- If *NO is specified on the RSTD parameter and *DEC is specified on the TYPE parameter and the RANGE parameter is specified, the range of possible values is displayed in the possible values field. The resulting line will appear in the form: a-b, *XXX, *YYY, *ZZZ (where a and b are numerals defining the range).
- If *YES is specified on the RSTD parameter, the possible values displayed are determined by the VALUES parameter, the SNGVAL parameter, and the SPCVAL parameter. The resulting line will appear in the form: *XXX, *YYY, *ZZZ...

***NONE:** No values are displayed.

***PGM:** A program that is called determines the values that are displayed. The program that is called is identified in CHOICEPGM parameter.

message-identifier: Specify the message ID of the message used to retrieve the message containing the text for the possible values field. The message file specified on the PMTFIELD parameter of the Create Command (CRTCMD) command is used to find the message.

'choices-text': Specify no more than 30 characters, enclosed in apostrophes.

CHOICEPGM Parameter

Specifies the qualified name of the program that is called during the prompting to fill in the possible choices text and the permissible values during prompting. This parameter must be specified if CHOICE(*PGM) is specified, and may not be specified otherwise.

***NONE:** No program is identified to fill in the possible choices text and permissible values.

The possible library values are:

***LIBL:** The library list is used to locate the program name.

***CURLIB:** The current library for the job is used to locate the program name. 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 where the program name is located.

program-name: Specify the name of the program to be called during prompting to fill in the possible choices text or a permissible value.

If an exception occurs when the program is called, no possible choices text is left blank, and the list of permissible values is taken from the command.

INLPMTLEN Parameter

Specifies the length of the input field initially displayed for the qualifier when the command is prompted. The user can extend the field to a maximum length of 512 bytes by entering an ampersand (&) in the first position of the field, followed by a blank. INLPMTLEN is valid only if TYPE is specified as *NAME, *SNAME, *CNAME, *GENERIC, or *CHAR. If FULL(*YES), RSTD(*YES), or CONSTANT are specified, INLPMTLEN(*CALC) must be specified or defaulted.

***CALC**: The prompter will determine the length of the prompt field based on the type and length of the parameter.

***PWD**: If the current value of system value QPWDLV is '0' or '1', the prompt field will be 10 bytes long. Otherwise, the length of the prompt field will be determined by the length of the parameter. INLPMTLEN(*PWD) is valid only if type is specified as *CHAR, *NAME, *SNAME, *PNAME, or *CNAME.

initial-prompt-length: Specify the initial length in bytes of the prompt field. Valid values are 1-12, 17, 25, 32, 50, 80, 132, 256, and 512.

PROMPT Parameter

Specifies the prompt text, if any, that is used for the qualifier (defined in this QUAL statement). The PROMPT parameter is not allowed for the first qualifier or for a qualifier for which the CONSTANT parameter is specified. The prompt text for the first qualifier comes from the PARM or ELEM statement PROMPT parameter that points to the qualifier. The prompt text describes the qualifier input field to the user, who may enter a response to the information displayed.

***NONE**: No prompt text is shown for the qualifier defined by this QUAL statement. This qualifier is still prompted by an input field, but no text is shown with it.

message-identifier: Specify the message identifier that specifies the message containing the prompt text of up to 30 characters that is shown. When the program is prompting for the qualifier. If a message having the specified identifier cannot be found in the message file specified in the PMTFILE parameter of the Create Command (CRTCMD) command, the message identifier itself is used as the prompt text.

'prompt-text': Specify the prompt text that is shown when the program is prompting the qualifier. The text must be a character string of no more than 30 characters, enclosed in apostrophes.

Example: QUAL statement

Example 1: Defining a SPLFILE Parameter

```
.-*-----<br>>>-SPLFILE(--file-name--+-----+>>-<br>          '-+-----+--job-name--'<br>          '-+-----+--user-name/-'<br>          '-job-number/-'<br><br>>-----<
```

Command definition statements:

```

      PARM KWD(SPLFILE) TYPE(L1) DFT(*) SNGVAL(*)
L1:   ELEM TYPE(*NAME) MIN(1) /*For file name */
      ELEM TYPE(Q1) /*For job name */
Q1:   QUAL TYPE(*NAME) MIN(1) /*For job name */
      QUAL TYPE(*NAME) /*For user name */
      QUAL TYPE(*CHAR) LEN(6) /*For job number */

```

The SPLFILE parameter is optional and, if not specified, defaults to an asterisk (*). Otherwise, the value consists of a two-element list. The first element is a file name and it is required. The second element is a qualified job name. The first qualifier is required; the last two qualifiers are optional.

Example 2: Defining a DTAMBRs Parameter

```

      .-*ALL-----|
      | .-----|
      | v .-*CURLIB/----- .-*NONE----- (1) |
>>-DTAMBRs(+-----library-name/+-----physical-file-name-+-----member-name-+-----)->
>-----<

```

Note:

1. A maximum of 32 repetitions

Command definition statements:

```

      PARM KWD(DTAMBRs) TYPE(L1) DFT(*ALL) MAX(32) +
      SNGVAL(*ALL)
L1:   ELEM TYPE(Q1) MIN(1)
      ELEM TYPE(*NAME) MIN(0) MAX(32) SPCVAL(*NONE) +
      DFT(*NONE)
Q1:   QUAL TYPE(*NAME) MIN(1)
      QUAL TYPE(*NAME) DFT(*CURLIB) SPCVAL(*CURLIB)

```

The parameter named DTAMBRs is optional and, if not specified, defaults to *ALL. Otherwise, the value consists of a list, each element of which is itself a list. Each sublist consists of a qualified file name optionally followed by one or more member names. If no member name is specified, *NONE is taken as the default. If no library qualifier is specified for the physical file, *CURRENT is taken as the default. This means that the library is the one currently indicated by the qualified physical file name saved in the description of the logical file to which this DTAMBRs parameter applies. Each sublist can contain one file name and up to 32 member names. Up to 32 such sublists can appear as the value of DTAMBRs.

Parameter values used for testing and debugging

This section contains expanded descriptions of the program variable, basing pointer, subscript, and qualified-name parameter values. These values can be specified on the Add Breakpoint (ADDBKP), Add Trace (ADDTRC), Change High-Level Language Pointer (CHGHLTPTR), Change Program Variable (CHGPGMVAR), and Display Program Variable (DSPPGMVAR) commands.

For expanded descriptions, see the following:

- “Program-variable description” on page 134
- “Basing-pointer description” on page 135
- “Subscript description” on page 135
- “Qualified-name description” on page 136

Program-variable description

Program Variable

```

      .-----|
      | v ----- (1) |
>>-qualified-name---subscript-----,subscript-----+-----<

```

Note:

1. A maximum of 14 repetitions

The program variable must be enclosed in apostrophes if it contains special characters. Up to 132 characters can be specified for a program variable name. This includes any subscripts, embedded blanks, parentheses, and commas. It does not include the enclosing apostrophes when special characters are used. Some examples are:

```
COUNTA
'VAR1(2,3)'
'A.VAR1(I,3,A,J,1)'
'VAR1 OF A(I,3,J OF A)'
'&LIBNAME'
```

Basing-pointer description

Basing Pointer

```

          .-----+----->
          |          |
          v          (1) |
>>-qualified-name---subscript-----,subscript-----+-----<<
```

Note:

1. A maximum of 14 repetitions

The basing pointer must be enclosed in apostrophes if it contains special characters. Up to 132 characters can be specified for a basing pointer name. This includes any subscripts, embedded blanks, parentheses, and commas. It does not include the enclosing apostrophes when special characters are used. Some examples are:

```
PTRVAR1
'ABC.PGMPTR(5,B,I)'
```

If more than one basing pointer is specified for a variable, the list of basing pointers must be enclosed in parentheses. When multiple basing pointers are specified, they must be listed in order, from the first basing pointer to the last, when used to locate the variable. In the example below, the PTR_1 basing pointer is the first basing pointer used to locate the variable; it either must have a declared basing pointer, or it must not be a based variable. The address contained in the PTR_1 pointer is used to locate the A.PTR_2 pointer (which must be declared as a based pointer variable). The contents of the A.PTR_2 pointer are used to locate the PTR_3 pointer array (which must also be declared based), and the contents of the specified element in the last pointer array are used to locate the variable. An example is:

```
('PTR_1' 'A.PTR_2' 'PTR_3(1,B,J)')
```

Subscript description

Subscript

```

.-integer-number--
>>-+-qualified-name--+-+-----><
'|_*-----'
```

An integer number contains from 1 through 15 digits with an optional leading sign (either plus or minus). A decimal point is not allowed in an integer-number subscript. If a decimal point is specified, the subscript value is not interpreted as the correct numeric value by the system, and an error message is returned.

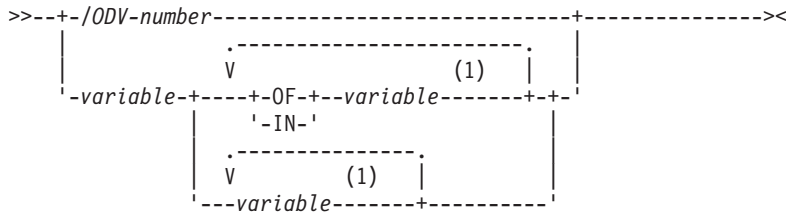
An asterisk (*) can be used to request a single-dimensional cross-section display of an array program variable. An asterisk can only be specified for a subscript on the primary variable (not on a basing pointer) for the PGMVAR keyword on the Add Break Point (ADDBKP), Add Trace (ADDTRC), and Display Program Variable (DSPPGMVAR) commands. In addition, if the variable has multiple dimensions, only one of the subscript values can be an asterisk. An example of a request to display an array cross-section is:

```
DSPPGMVAR PGMVAR('X1(*,5,4)')
```

This display shows the values of all elements of the array that have the second subscript equal to five, and the third subscript equal to four.

Qualified-name description

Qualified-Name



Note:

1. A maximum of 19 repetitions

Some high-level languages may allow you to declare more than one variable with the same fully qualified name (although you generally are not able to refer to these variables in the high-level language program after they are declared). If you attempt to refer to such a variable using an OS/400 test facility command, the system selects one of the variables and uses it for the operation. No error is reported when a duplicate fully qualified name is selected.

Rules for qualified name description

- An ODV number is a slash (/) followed by 1 to 4 hexadecimal digits (0 through 9, and A through F).
- The variable-name must be the name of a variable in the program. This name must be specified the same way in the high-level language. Some high-level languages introduce qualifier variable names in addition to the ones you specified in the source for your program. See the appropriate high-level language manual for more information about variable names.
- Blanks must separate the variable-names from the special words OF and IN.
- When a period is used to form a qualified name, no blanks can appear between it and the variable-names.
 - The ordering of the variable names must follow these rules:
 - For qualified names that contain no embedded period, the variable names are assumed to be specified from the lowest to the highest levels in the structure.
 - For qualified names that contain one or more embedded periods, the variable names are assumed to be specified from the highest to the lowest levels in the structure.
- When an ODV number is not used for the qualified name, enough qualifier variable names must be specified so that a single unique variable can be identified in the program. Whether the qualified name is a simple name (only one variable name specified) or a name with multiple qualifier variable names, the variable in the program is uniquely identified if either of the following conditions is true (these conditions may require you to specify more qualifier variable names for OS/400 test facility commands than you need to specify in the high-level language program to uniquely select a program variable):
 - A variable is uniquely identified if there is one and only one variable in the program with a set of qualifier variables matching the qualified variable name specified.
 - A variable is uniquely identified in the program if it has exactly the same set of qualifier variables as the qualifier variable names specified. When the complete set of qualifiers is specified, the variable name is said to be *fully qualified*. A variable that is a *fully qualified* match for the qualified-name is selected even if there are other variables with names that match the qualified name but have additional qualifier variables which were not specified.

OS/400 objects

Operating System/400* (OS/400*) objects provide the means through which all data processing information is stored and processed by the iSeries 400. An **OS/400 object** is a named unit that exists (occupies space) in storage and on which operations are performed on it by the operating system.

CL commands perform operations on the Operating System/400* (OS/400*) objects. Several types of OS/400 objects are created and used in the control language.

OS/400 objects have the following in common:

- Objects have a set of descriptive attributes that are defined when the object is created.
- Objects have to be used by the system to perform a specific function must be specified in the CL command that performs that function.
- Objects have a set of attributes that describe it and give the specific values assigned for those attributes.
- Generally, objects are independent of all other objects. However, some objects must be created before other objects can be created; for example, a logical file cannot be created if the physical file it must be based on does not exist.
- Objects must be created before other operations that use the object are performed. Descriptions of the create commands (those commands that begin with the letters CRT) give more information about the object types that they create.
- Every OS/400 object that is used by the control language has a name. The object name specified in a CL command identifies which object is used by the operating system to perform the function of the command.
- Objects have either a simple, qualified, or generic name.

For additional information about OS/400 objects, see the following:

- “Library objects”
- “Object types”
- “Simple and qualified object names” on page 139
- “Generic object names” on page 140
- “Object naming rules” on page 141
- Predefined Values and Default Library Locations for OS/400 Object Types table (on page 137)

Library objects

Many objects are grouped in special objects called libraries. “Object types” provides information about various object types and their default libraries.

Some objects, which use the integrated file system, are located in directories and can be found by using path names or object name patterns instead of searching libraries. You can also use these directories to locate objects. For more information about integrated file system commands, see the Integrated File Systems topic.

Object types

The following table lists the predefined values for all the OS/400 object types. When an object is created and a library qualifier can be specified but is not, the object is stored in the user’s current job library, as shown in the last column. The user profile for each user specifies the user’s current library. The current library will be QGPL if it is not specified otherwise. The other objects, identified by N/A in the last column, cannot be stored in user-provided libraries.

Table 4. Predefined Values and Default Library Locations for OS/400 Object Types

Value	Object Type	Default User Library
*ALRTBL	Alert table	*CURLIB

Value	Object Type	Default User Library
*AUTL	Authorization list	N/A
*BLKSF	Block special file	N/A
*BNDDIR	Binding directory	*CURLIB
*CFGL	Configuration list	N/A
*CHRSF	Character special file	N/A
*CHTFMT	Chart format	*CURLIB
*CLD	C/400 locale description	*CURLIB
*CLS	Class	*CURLIB
*CMD	Command	*CURLIB
*CNL	Connection list	QSYS
*COSD	Class-of-service description	QSYS
*CRG	Cluster resource group	N/A
*CRQD	Change request description	*CURLIB
*CSI	Communications Side Information	*CURLIB
*CTLD	Controller description	QSYS
*DDIR	Distributed file directory	N/A
*DEVD	Device description	QSYS
*DIR	Directory	N/A
*DOC	Document	QDOC
*DSTMF	Distributed stream file	N/A
*DTAARA	Data area	*CURLIB
*DTADCT	Data dictionary	N/A
*DTAQ	Data queue	*CURLIB
*EDTD	Edit description	QSYS
*EXITRG	Exit registration	N/A
*FCT	Forms control table	*CURLIB
*FIFO	First-in-out special file	N/A
*FILE	File	*CURLIB
*FLR	Folder	QDOC
*FNTRSC	Font resources	*CURLIB
*FNTTBL	Font mapping table	*CURLIB
*FORMDF	Form definition	*CURLIB
*FTR	Filter	*CURLIB
*GSS	Graphics symbol set	*CURLIB
*IGCDCT	Double-byte character set (DBCS) conversion dictionary	*CURLIB
*IGCSRT	Double-byte character set (DBCS) sort table	*CURLIB
*IGCTBL	Double-byte character set (DBCS) font table	N/A
I *IMGCLG	Image catalog	N/A
*IPXD	Internetwork packet exchange description	QSYS
*JOB	Job description	*CURLIB
*JOBQ	Job queue	*CURLIB
*JOBSCD	Job schedule	*CURLIB
*JRN	Journal	*CURLIB
*JRNRCV	Journal receiver	*CURLIB
*LIB	Library	QSYS
*LIND	Line description	QSYS
*MENU	Menu description	*CURLIB
*MGTCOL	Management collection	QPFRDATA
*MODD	Mode description	QSYS
*MODULE	Compiler unit	*CURLIB
*MSGF	Message file	*CURLIB
*MSGQ	Message queue	*CURLIB
*M36	AS/400 Advanced 36 machine	*CURLIB
*M36CFG	AS/400 Advanced 36 machine configuration	*CURLIB

Value	Object Type	Default User Library
*NODGRP	Node group	*CURLIB
*NODL	Node list	*CURLIB
*NTBD	NetBIOS description	QSYS
*NWID	Network interface description	QSYS
*NWSD	Network server description	QSYS
*OUTQ	Output queue	*CURLIB
*OVL	Overlay	*CURLIB
*PAGDFN	Page definition	*CURLIB
*PAGSEG	Page segment	*CURLIB
*PDG	Print Descriptor Group	*CURLIB
*PGM	Program	*CURLIB
*PNLGRP	Panel group definition	*CURLIB
*PRDAVL	Product availability	QSYS
*PRDDFN	Product definition	QSYS
*PRDLOD	Product load	QSYS
*PSFCFG	Print Services Facility configuration	*CURLIB
*QMFORM	Query management form	*CURLIB
*QMQRV	Query management query	*CURLIB
*QRYDFN	Query definition	QGPL
*RCT	Reference code translate table	QGPL
*SBSD	Subsystem description	*CURLIB
*SCHIDX	Search index	QGPL
*SOCKET	Local socket	N/A
*SPADCT	Spelling aid dictionary	QGPL
*SRVPGM	Service program	*CURLIB
*SQLPKG	Structured Query Language package	*CURLIB
*SQLUDT	User-defined SQL type	N/A
*SSND	Session description	QGPL
*STMF	Bytestream file	N/A
*S36	System/36 machine description	QGPL
*SVRSTG	Server storage space	N/A
*SYMLNK	Symbolic link	N/A
*TBL	Table	*CURLIB
*USRIDX	User index	*CURLIB
*USRPRF	User profile	QSYS
*USRQ	User queue	*CURLIB
*USRSPC	User space	*CURLIB
*VLDL	Validation list	*CURLIB
*WSCST	Work station customizing object	*CURLIB

Simple and qualified object names

The name of a specific object that is located in a library can be specified as a simple name or as a qualified name. A *simple object name* is the name of the object only. A *qualified object name* is the name of the library where the object is stored followed by the name of the object. In a qualified object name, the library name is connected to the object name by a slash (/).

Either the simple name or the qualified name of an object can be specified if the object exists in one of the libraries named in the job's library list; the library qualifier is optional in this case. A qualified name *must* be specified if the named object is not in a library named in the library list.

Note: Although a job name also has a qualified form, it is not a qualified object name because a job is not an OS/400 object. A job name is qualified by a user name and a job number, not by a library name. For more information about the JOB parameter, refer to JOB parameter.

Example: The following table shows how simple and qualified object names are formed.

Name Type	Name Syntax	Example
Simple object name	object-name	OBJA
Qualified object name	library-name/object-name	LIB1/OBJB

Generic object names

A *generic object name* may refer to more than one object. That is, a generic name contains one or more characters that are the first group of characters in the names of several objects; the system then searches for all the objects that have those characters at the beginning of their names and are in the libraries named in the library list. A generic name is identified by an asterisk (*) as the last character in the name.

A quoted generic name consists of a generic name enclosed in quotation marks. Unlike quoted names, if there are no special characters between the quotation marks, the quotation marks are not removed. The generic name "ABC*" would cause the system to search for objects whose name begins with "ABC".

A generic name can also be qualified by a library name. If the generic name is qualified, the system will search only the specified library for objects whose names begin with that generic name.

Note: A generic name also can be qualified by one or more directories if it is a path name. In a path name, letters can be specified before and after the asterisk (*). For more information on path names, see the Integrated File System topic.

When you specify a generic name, the system performs the required function on all objects whose names begin with the specified series of characters. You must have the authority required to perform that function on every object the generic name identifies. If you do not have the required authority for an object, the function is not performed and a diagnostic message is issued for each instance that the attempted generic function failed. A completion message is issued for each object the generic function operates on successfully. You must view the online low-level messages to see the completion messages. Once the entire generic function is completed, a completion message is issued that states that all objects were operated on successfully. If one or more objects could not be successfully operated on an escape message is issued. If an override is in effect for a specific device file, the single object name specified on the override, rather than the generic name, is used to perform the operation.

You may not be able to use a generic name for delete, move, or rename commands if the library containing the objects is already locked. A search for generic object names requires a more restrictive lock on the library containing the objects than a search for full object names. The more restrictive lock is necessary to prevent another user from creating an object with the same name as the generic search string in the library while the delete, move, or rename command is running. You can circumvent this problem by using the full name of the objects instead of a generic name. Or you can end the job or subsystem that has a lock on the library.

Note: Use the WRKOBJLCK (Work with Object Locks) command to determine which jobs or subsystems have a lock on the library.

For some commands, a library qualifier can be specified with the generic name to limit the scope of the operation. For example, a Change Print File (CHGPRTF) command with FILE(LIB1/PRT*) performs the desired operation on printer files beginning with PRT in library LIB1 only; printer files in other libraries are not affected.

The limitations associated with the various library qualifiers are as follows:

- *library-name*: The operation is performed on generic object names only in the specified library.
- *LIBL: The operation is performed on generic object names in the library list associated with the job that requested the generic operation.
- *CURLIB: The operation is performed on generic object names in the current library.

- ***ALL:** The operation is performed on generic object names in all libraries on the system for which you are authorized.
- ***USRLIBL:** The operation is performed on generic object names in the user part of the library list for the job.
- ***ALLUSR:** The operation is performed on all nonsystem libraries (libraries that do not start with the letter Q), except for the following:

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

The operation is also performed on the following Qxxx libraries:

```
QDSNX      QRCL       QUSRIJS    QUSRSYS
QGPL       QS36F     QUSRINFSCR QUSRVxRxMx
QGPL38     QUSER38   QUSRNOTES
QMPGDATA   QUSRADSM  QUSRPOSGS
QMOMDATA   QUSRBRM   QUSRPOSSA
QMOMPROC   QUSRDIRCL QSRPYMSVR
QPFRDATA   QUSRDIRDB QSRRDARS
```

Note: 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.

Example: Generic object name:

Name Type	Name Syntax	Example
Simple generic name	generic-name*	OBJ*
Qualified generic name	library-name/generic-name*	LIB1/OBJ*
Quoted generic name	"generic-name**"	"ABC**"

Object naming rules

The following rules, in addition to the standard rules for all names, are used to name all OS/400 objects used in control language commands. The syntax diagram for each CL command shows whether a simple object name, a qualified name, or a generic name can be specified.

- **Naming a Single Object:** In the name of a single object, each part (the simple name and the library qualifier name) can have a maximum of 10 characters. For more information on specifying objects, see Generic object names.
- **Naming a User-Created Object:** To distinguish a user-created object from an IBM-supplied object, you should not begin user-created object names with Q because the names of all IBM-supplied objects (except commands) begin with Q. Although you can use as many as 10 characters in CL object names, you may need to use fewer characters to be consistent with the naming rules of the particular high-level language that you are also using. Also, the high-level language might not allow underscores in the naming rules. For example, RPG limits file names to eight characters and does not allow underscores.
- **Naming a Generic Object:** In a generic name, a maximum of nine alphanumeric characters can be used, not including the asterisk (*) that must immediately follow the last character. For more information on using generic names, see "Generic object names" on page 140.

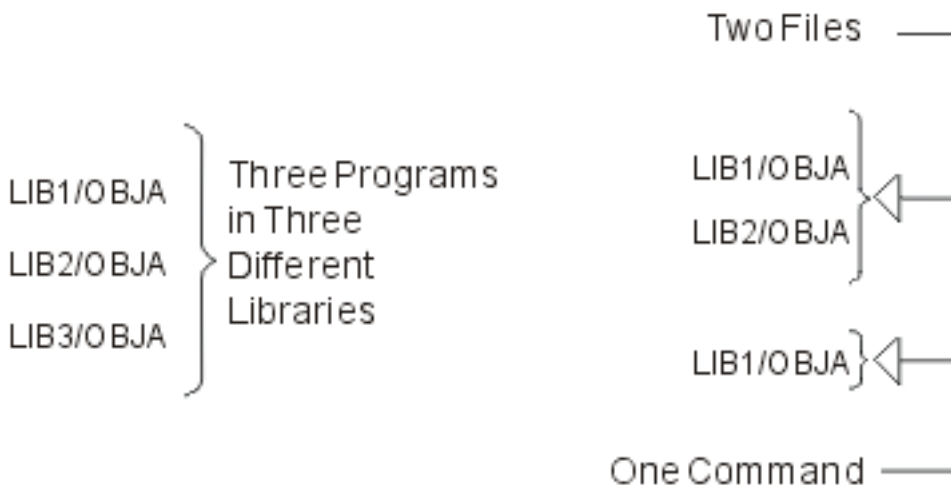
INV and INV* are valid values where a generic name is accepted. When the name INV is specified, only the object INV is referenced. When the quoted generic name INV* is specified, objects that begin with INV are referred to, such as INV, INVOICE, INVENTORY, and INVENPGM1. When the quoted generic name "INV*" is specified, objects that begin with "INV are referred to, such as "INV%1" and "INV>.

- **Object Library Qualifier Limitations:** No library qualifier can be specified with the object name if the object being created is a library, user profile, line description, controller description, device description, mode description, class-of-service description, or configuration list. A library name can never be qualified because a library cannot be placed in a library. The other object types (*USRPRF, *LIND, *CTLD, *DEVD, *MODD, *COSD, and *CFGL) appear to be types that exist only in the QSYS library. When only

the name of an object of these object types is accepted, a library qualifier cannot be specified with the object name. On the Display Object Description (DSPOBJD) command, where any object name is accepted, QSYS can be specified.

- *Library List Qualifiers:* The predefined value *LIBL (and others, such as *CURLIB and *ALLUSR) can be used in place of a library name in most commands. *LIBL indicates that the libraries named in the job's library list are used to find the object named in the second part of the qualified name.
- *Duplicate Object Names:* Duplicate names are not allowed for objects of the same type in the same library.

Two objects with the same name cannot be stored in the same library unless their object types are different. Two objects named OBJA can be stored in the library LIBx only if, for example, one of the objects is a program and the other is a file. The following combinations of names and object types could all exist on the system at the same time.



If more than one library contains an object with the same name (and both libraries are in the same library list) and a library qualifier is not specified with the object name, the first object found by that name is used. Therefore, when you have multiple objects with the same name, you should specify the library name with the object name or ensure that the appropriate library occurs first in the library list. For example, if you are testing and debugging and choose not to qualify the names, ensure that your test library precedes your production library in the library list.

Default libraries

In a qualified object name, the library name is always optional. If a library name is not specified, the default given in the command's description is used (usually either *CURLIB or *LIBL). If the named object is being created, the current library is the default; when the object is created, it is placed either in the current library or in the QGPL (the general purpose library) if no current library is defined. For objects that already exist, *LIBL is the default for most commands, and the job's library list is used to find the named object. The system searches all of the libraries currently in the library list until it finds the object name specified.

For additional information about object naming, see the following:

- "Path names (*PNAME)" on page 143
- "Generic names (*GENERIC)" on page 144
- "Additional rules for unique names" on page 144

Related information:

- “Naming within commands” on page 74
- “Folder and document names” on page 76

Path names (*PNAME): A path name is a character string that can be used to locate objects in the integrated file system. The string can consist of one or more elements, each separated by a slash (/) or back slash (\). Each element is typically a directory or equivalent, except for the last element, which can be a directory, another object such as a file, or a generic presentation of an object or objects to be located.

The / and \ characters and nulls cannot be used in the individual components of the path name because the / and \ characters are used as separators. The name may or may not be changed to uppercase, depending on whether the file system containing the object is case-sensitive and whether the object is being created or searched for. If the parameter is defined as CASE(*MONO) (the default), any values that are not enclosed in single quotes will be changed to uppercase by the command analyzer.

A / or \ character at the beginning of a path name means that the path begins at the top most directory, the “root” (/) directory. If the path name does not begin with a / or \ character, the path is assumed to begin at the current directory of the user entering the command.

The path name must be represented in the CCSID currently in effect for the job. If the CCSID of the job is 65535, the path name must be represented in the default CCSID of the job. Hard-coded path names in programs are encoded in CCSID 37. Therefore, the path name should be converted to the job CCSID before being passed to the command. The maximum length of the path name character string on the CL commands is 5000 characters.

When operating on objects in the QSYS.LIB file system, the component names must be of the form name.object-type; for example:

```
'/QSYS.LIB/PAY.LIB/TAX.FILE'
```

Path names must be enclosed in apostrophe (') marks when entered on a command line if they contain special characters. These marks are optional when path names are entered on displays. If the path name includes any quoted strings or special characters; however, the enclosing " marks must be included. The following are rules for using special characters:

- A tilde (~) character followed by a slash or backslash at the beginning of a path name means that the path begins at the home directory of the user entering the command.
- A tilde (~) character followed by a user name and then a slash or backslash at the beginning of a path name means that the path begins at the home directory of the user identified by the user name.
- In some commands, an asterisk (*) or a question mark (?) can be used in the last component of a path name to search for patterns of names. The * tells the system to search for names that have any number of characters in the position of the * character. The ? tells the system to search for names that have a single character in the position of the ? character.

- To avoid confusion with iSeries 400 special values, path names cannot start with a single asterisk (*) character. To perform a pattern match at the beginning of a path name, use two asterisks (**).

Note: This only applies to relative path names where there are no other characters before the asterisk.

- The path name must be enclosed in apostrophes (') or quotation marks (") if any of the following characters are used in a component name:
 - Asterisk (*)
 - Question mark (?)
 - Apostrophe (')
 - Quotation mark (")
 - Tilde (~), if used as the first character in the first component name of the path name (if used in any other position, the tilde is interpreted as just another character)

This practice is not recommended because the meaning of the character in a command string could be confused and it is more likely that the command string will be entered incorrectly.

- Do not use a colon (:) in path names. It has a special meaning within the system.
- The processing support for commands and associated user displays does not recognize code points below hexadecimal 40 as characters that can be used in command strings or on displays. If these code points are used, they must be entered as a hexadecimal representation, such as the following:

```
crtmdir dir(X'02')
```

Therefore, use of code points below hexadecimal 40 in path names is not recommended. This restriction applies only to commands and associated displays, not to APIs.

- | For further information on device names, see the Specifying the device name article in the Systems management, Backup and recovery topic.

For further information on path names, see the Integrated File System topic in the Information Center.

Generic names (*GENERIC): A generic name is one that contains at least one initial character that is common to a group of objects, followed by an asterisk. (The asterisk identifies the series of common characters as a generic name; otherwise, the system interprets the series of characters as the name of a specific object).

For more information on *GENERIC names, see the Generic object names section in this topic.

Additional rules for unique names: Additional rules involving special characters (as an extra character) that apply to the following types of names are:

- A *command label* must be immediately followed by a colon (:). Blanks can follow the colon, but none can precede it. A command label name cannot be a quoted name.
- A *CL variable name* must be preceded by an ampersand (&) to indicate that it is a CL variable used in a CL program.
- A *built-in function name* must be preceded by a percent sign (%) to indicate that it is an IBM-supplied built-in function that can be used in an expression. A built-in function name cannot be a quoted name.

These special characters are not part of the name; each is an additional character attached to a name (making a maximum of 11 characters) indicating to the system what the name identifies.

The names of OS/400 objects, CL program variables, system values, and built-in functions can be specified in the parameters of individual commands as indicated in the syntax diagram for each command. Instead of specifying a constant value, a CL variable name can be used on most parameters in CL programs to specify a value that may change during the running of programs. It is the contents of the variable that identify the objects and variables that are used when the command is run.

Commonly used parameters: Expanded descriptions

This section contains the expanded descriptions of some of the parameters commonly used in the CL commands. The parameters included here meet one or both of these criteria:

- There is extensive information about how they are used.
- They are used in many of the CL commands (such as the AUT parameter), and the parameter description in the individual command description gives only the essential information.

The expanded descriptions of the applicable command parameters have been placed here for several reasons:

- to reduce the amount of material needed in the individual commands. Normally programmers familiar with a parameter's main function do not need the details.
- to provide the supplemental information that is useful to programmers in some instances.

The format for this information is designed for easy reference and includes a general description of each parameter that explains its function, states the rules for its use, and provides other helpful information. The values that can be specified for each parameter are also listed. Each value is followed by an explanation of what it means and (possibly) in which commands it is used. Not all of the values appear in every command. Refer to the individual command descriptions for the specific use of the value in that command parameter.

Double-byte character text in CL commands

AUT parameter	OBJ parameter
CLS parameter	OBJTYPE parameter
COUNTRY parameter	OUTPUT parameter
EXCHTYPE parameter	PRTTXT parameter
FILETYPE parameter	REPLACE parameter
FRCRATIO parameter	Scheduling priority parameters (JOBPTY, OUTPTY, PTYLMT)
IGCFEAT parameter	SEV parameter
JOB parameter	SPLNBR parameter
LABEL parameter	TEXT parameter
MAXACT parameter	VOL parameter
	WAITFILE parameter

Double-byte character text in CL commands

You can use double-byte character data anywhere in a CL command that descriptive text can be used.

Enter double-byte character text as follows:

1. Begin the double-byte character text with an apostrophe (').
2. Enter a shift-out character.
3. Enter the double-byte character text.
4. Enter a shift-in character.
5. End the double-byte character text with an apostrophe (').

For example, to enter the double-byte character literal ABC, enter the following, where 0_E represents the shift-out character and 0_F represents the shift-in character:


```
'0EABC0F'
```

Limit the length of a double-byte character text description of an object to 14 double-byte characters, plus the shift control characters, to make sure that the description is properly displayed and printed.

AUT parameter

You use the authority (AUT) parameter in create, grant, and revoke commands. It specifies the authority granted to all users of an object. It also specifies an authorization list that is used to secure the object. Four object types allow the AUT parameter to contain an authorization list: LIB, PGM, DTADCT, and FILE. Public authority is an OS/400 object attribute that controls the base set of rights to that object for all users having access to the system. These rights can be extended or reduced for specific users. If you specify an authorization list, the public authority in the authorization list is the public authority for the object. The owner of an object has all authority to the object at its creation.

If the object is created as a private object or with the limited authority given to all users, the owner can grant more or less authority to specific users by specifically naming them and stating their authority in the Grant Object Authority (GRTOBJAUT) command. The owner also can withdraw specific authority from specific users, or from all users (publicly authorized and/or specifically authorized) by using the Revoke Object Authority (RVKOBJAUT) command or the Edit Object Authority (EDTOBJAUT) command.

The iSeries Security Reference  book has a complete description of security provisions and applicable rights of use by object type.

Values allowed

***LIBCRTAUT:** The public authority for the object is taken from the value on the CRTAUT parameter of the target library (the library that is to contain the object). The public authority is determined when the object is created. If the CRTAUT value for the library changes after the object is created, the new value does not affect any existing objects.

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

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

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

***EXCLUDE:** You cannot access the object.

***EXECUTE:** You can run a program or procedure or search a library or directory.

authorization-list-name: Specify the name of the authorization list whose authority is used.

CLS parameter

The class (CLS) parameter identifies the attributes that define the run time environment of a job. The following attributes are defined in each class:

- **Run priority:** A number that specifies the priority level assigned to all jobs running that use the class. The priority level is used to determine which job, of all the jobs competing for system resources, is run next. The value can be 1 through 99, where 1 is the highest priority (all jobs having a 1 priority are run first).
- **Time slice:** The maximum amount of processor time that the system allows the job to run when it is allowed to begin. The time slice indicates the amount of time needed for the job to accomplish a meaningful amount of work (the time used by the system for reading auxiliary storage is not charged against the time slice). When the time slice ends, the job waits while other queued jobs of the same or higher priority are allowed to run (up to the time specified in their time slices); then the job is given another time slice.
- **Purge value:** Indicates whether the job step is eligible to be moved from main storage to auxiliary storage while the job is waiting for some resource before it can continue, or when its time slice is used up and equal or higher priority jobs are waiting.
- **Default wait time:** The default amount of time that the system waits for the completion of an instruction that performs a wait. This wait time applies to times when an instruction is waiting for a system action, not to the time an instruction is waiting for a response from a user. Normally, this would be the amount of time you are willing to wait for the system before ending the request. If the wait time is exceeded, an error message is passed to the job. This default wait time applies only when a wait time is not specified in the CL command that causes the wait.

The wait time used for allocating file resources is specified in the file description and can be overridden by an override command. It specifies that the wait time specified in the class object is used. If file resources are not available when the file is opened, the system waits for them until the wait time ends.

Note: The class attributes apply to each routing step of a job. Most jobs have only one routing step, but if the job is rerouted (because of something like the Remote Job or Transfer Job command) the class attributes will be reset.

- **Maximum CPU time:** The maximum amount of processor time (the sum of the time slices if they differ, or time slice period multiplied by number of time slices if they are equal) allowed for a job's routing step to complete processing. If the job's routing step is not completed in this amount of time, it is ended, and a message is written to the job log.
- **Maximum temporary storage:** The maximum amount of temporary storage that can be used by a job's routing step. This temporary storage is used for the programs that run in the job, for the system objects used to support the job, and for temporary objects created by the job.

The system is shipped with a set of classes that define the attributes for several job processing environments. Other classes can be created by the Create Class (CRTCLS) command; any class can be displayed or deleted by the respective Display Class (DSPCLS) and Delete Class (DLTCLS) commands.

Values allowed

qualified-class-name: Specify the name of the class, optionally qualified by the name of the library in which the class is stored. If the class name is not qualified and the CLS parameter is in the CRTCLS command, the class object is stored in *CURLIB; otherwise, the library list (*LIBL) is used to find the class name.

The following classes (by name) are supplied with the system:

QGPL/QBATCH

For use by batch jobs

QSYS/QCTL

For use by the controlling subsystem

QGPL/QINTER

For use by interactive jobs

QGPL/QPGMR

For use by the programming subsystem

QGPL/QSPL

For use by the spooling subsystem printer writer

QGPL/QSPL2

For general spooling use in the base system pool

COUNTRY parameter

The Country parameter specifies the country or region code part of the X.400 O/R name. An ISO 3166 Alpha-2 code or an ITU-T country or region code can be specified. (The ITU-T country or region code is the data country or region or geographical area code published in the "International Numbering Plan for Public Data Networks," Recommendation X.121 (09/92), by the ITU-T (formerly CCITT). Table 4 is a list of the possible country or region codes that can be specified.

Values allowed

***NONE:** No country or region code is specified.

country-code: Specify an ISO 3166 Alpha-2 code or a CCITT (also known as ITU-2) country or region code from the following table.

Table 6. ISO X.400 Country or Region Codes

Country or Region	ISO 3166 Alpha-2 Code	ITU-T ¹ Country or Region Code
Afghanistan	AF	412

Country or Region	ISO 3166 Alpha-2 Code	ITU-T ¹ Country or Region Code
Albania	AL	276
Algeria	DZ	603
American Samoa	AS	544
Andorra	AD	
Angola	AO	631
Anguilla	AI	
Antarctica	AQ	
Antigua and Barbuda	AG	344
Argentina	AR	722
Armenia	AM	283
Aruba	AW	362
Australia	AU	505
Austria	AT	232
Azerbaijan	AZ	400
Bahamas	BS	364
Bahrain	BH	426
Bangladesh	BD	470
Barbados	BB	342
Belarus	BY	257
Belgium	BE	206
Belize	BZ	702
Benin	BJ	616
Bermuda	BM	350
Bhutan	BT	
Bolivia	BO	736
Bosnia and Herzegovina	BA	
Botswana	BW	652
Bouvet Island	BV	
Brazil	BR	724
British Indian Ocean Terr.	IO	
Brunei Darussalam	BN	528
Bulgaria	BG	284
Burkina Faso	BF	613
Burundi	BI	642
Cambodia	KH	456
Cameroon	CM	624
Canada	CA	302, 303
Cape Verde	CV	625
Cayman Islands	KY	346
Central African Republic	CF	623
Chad	TD	622
Chile	CL	730
China	CN	460
Christmas Island	CX	
Cocos (Keeling) Islands	CC	
Colombia	CO	732
Comoros	KM	654
Congo	CG	629
Cook Islands	CK	548
Costa Rica	CR	712
Cote d'Ivoire	CI	612
Croatia	HR	
Cuba	CU	368
Cyprus	CY	280

Country or Region	ISO 3166 Alpha-2 Code	ITU-T¹ Country or Region Code
Czech Republic	CZ	230
Denmark	DK	238
Djibouti	DJ	638
Dominica	DM	366
Dominican Republic	DO	370
East Timor	TP	
Ecuador	EC	740
Egypt	EG	602
El Salvador	SV	706
Equatorial Guinea	GQ	627
Eritrea	ER	
Estonia	EE	248
Ethiopia	ET	636
Falkland Islands (Malvinas)	FK	
Faroe Islands	FO	288
Fiji	FJ	542
Finland	FI	244
France	FR	208, 209
France, Metropolitan	FX	
French Antilles		340
French Guiana	GF	742
French Polynesia	PF	547
French Southern Terr.	TF	
Gabon	GA	628
Gambia	GM	607
Georgia	GE	282
Germany	DE	262 - 265
Ghana	GH	620
Gibraltar	GI	266
Greece	GR	202
Greenland	GL	290
Grenada	GD	352
Guadeloupe	GP	
Guam	GU	535
Guatemala	GT	704
Guinea	GN	611
Guinea-Bissau	GW	632
Guyana	GY	738
Haiti	HT	372
Heard and Mc Donald Islands	HM	
Honduras	HN	708
China (Hong Kong S.A.R.)	HK	453, 454
Hungary	HU	216
Iceland	IS	274
India	IN	404
Indonesia	ID	510
Iran	IR	432
Iraq	IQ	418
Ireland	IE	272
Israel	IL	425
Italy	IT	222
Jamaica	JM	338
Japan	JP	440 - 443
Jordan	JO	416

Country or Region	ISO 3166 Alpha-2 Code	ITU-T ¹ Country or Region Code
Kazakhstan	KZ	401
Kenya	KE	639
Kiribati	KI	545
Korea, Democratic People's Republic	KP	467
Korea, Republic of	KR	450, 480, 481
Kuwait	KW	419
Kyrgyzstan	KG	437
Lao People's Democratic Rep.	LA	457
Latvia	LV	247
Lebanon	LB	415
Lesotho	LS	651
Liberia	LR	618
Libyan Arab Jamahiriya	LY	606
Liechtenstein	LI	
Lithuania	LT	246
Luxembourg	LU	270
China (Macau S.A.R.)	MO	455
Macedonia ²	MK ²	
Madagascar	MG	646
Malawi	MW	650
Malaysia	MY	502
Maldives	MV	472
Mali	ML	610
Malta	MT	278
Marshall Islands	MH	
Martinique	MQ	
Mauritania	MR	609
Mauritius	MU	617
Mayotte	YT	
Mexico	MX	334
Micronesia	FM	550
Moldova, Republic of	MD	259
Monaco	MC	212
Mongolia	MN	428
Montenegro ²	ME ²	
Montserrat	MS	354
Morocco	MA	604
Mozambique	MZ	643
Myanmar	MM	414
Namibia	NA	649
Nauru	NR	536
Nepal	NP	429
Netherlands	NL	204, 205
Netherlands Antilles	AN	362
New Caledonia	NC	546
New Zealand	NZ	530
Nicaragua	NI	710
Niger	NE	614
Nigeria	NG	621
Niue	NU	
Norfolk Island	NF	
Northern Mariana Islands	MP	534
Norway	NO	242

Country or Region	ISO 3166 Alpha-2 Code	ITU-T ¹ Country or Region Code
Oman	OM	422
Pakistan	PK	410
Palau	PW	
Panama	PA	714
Papua New Guinea	PG	537
Paraguay	PY	744
Peru	PE	716
Philippines	PH	515
Pitcairn	PN	
Poland	PL	260
Portugal	PT	268
Puerto Rico	PR	330
Qatar	QA	427
Reunion	RE	647
Romania	RO	226
Russian Federation	RU	250, 251
Rwanda	RW	635
St. Helena	SH	
St. Kitts and Nevis	KN	356
St. Lucia	LC	358
St. Pierre and Miquelon	PM	308
St. Vincent and the Grenadines	VC	360
Samoa, Western	WS	549
San Marino	SM	292
Sao Tome and Principe	ST	626
Saudi Arabia	SA	420
Senegal	SN	608
Serbia ²	SP ²	
Seychelles	SC	633
Sierra Leone	SL	619
Singapore	SG	525
Slovakia	SK	
Slovenia	SI	
Solomon Islands	SB	540
Somalia	SO	637
South Africa	ZA	655
South Georgia and the S.S.I	GS	
Spain	ES	214
Sri Lanka	LK	413
Sudan	SD	634
Suriname	SR	746
Svalbard and Jan Mayen Is.	SJ	
Swaziland	SZ	653
Sweden	SE	240
Switzerland	CH	228
Syrian Arab Republic	SY	417
Taiwan	TW	466
Tajikistan	TJ	436
Tanzania, United Republic of	TZ	640
Thailand	TH	520
Togo	TG	615
Tokelau	TK	
Tonga	TO	539
Trinidad and Tobago	TT	374

Country or Region	ISO 3166 Alpha-2 Code	ITU-T ¹ Country or Region Code
Tunisia	TN	605
Turkey	TR	286
Turkmenistan	TM	438
Turks and Caicos Islands	TC	376
Tuvalu	TV	
Uganda	UG	641
Ukraine	UA	255
United Arab Emirates	AE	424, 430, 431
United Kingdom	GB	234, 235, 236, 237
United States	US	310 - 316
United States Minor Outlying Is.	UM	
Uruguay	UY	748
Uzbekistan	UZ	434
Vanuatu	VU	541
Vatican City State (Holy See)	VA	225
Venezuela	VE	734
Viet Nam	VN	452
Virgin Is. (Brit.)	VG	348
Virgin Is. (U.S.)	VI	332
Wallis and Futuna Is.	WF	543
Western Sahara	EH	
Yemen	YE	421, 423
Yugoslavia, territories of the former	YU	220
Zaire	ZR	630
Zambia	ZM	645
Zimbabwe	ZW	648

Note:

- ¹ This International Telecommunication Union (ITU) committee was formerly known as CCITT.
- ² At the time of publication, the ISO 3166 Alpha-2 Code for this country or region could not be confirmed. Before using this code, be sure to confirm with the latest ISO 3166 standard.

EXCHTYPE parameter

You use the Exchange Type (EXCHTYPE) parameter to specify which one of the three diskette exchange types (basic, H, or I) the system will use when writing diskette files. The exchange type will be stored in the volume label area on the diskette. There is one label for each data file on the diskette.

You can specify the exchange type to be used in one of the diskette device file commands (Create Diskette File (CRTDKTF), Change Diskette File (CHGDKTF), or Override Diskette File (OVRDKTF)). It can also be passed as a parameter when the device file is opened by the high-level language program (if supported).

The system will not use the exchange type specified in the diskette device file or high-level language program when processing a diskette input file. Instead, the system will use the exchange type from the file label on the diskette.

It is possible for one diskette to contain both basic and I exchange format files or H and I format files. One diskette cannot contain both basic and H format files.

Values allowed

You can specify one of the following values for the EXCHTYPE parameter, depending on the diskette type (1, 2, or 2D) and the diskette sector size (128, 256, 512, or 1024 bytes):

***STD:** The system determines the exchange type based on the diskette type and sector size. A basic exchange type is used if the diskette type is 1 or 2, and the diskette sector size is 128 bytes. An H exchange type is used if the diskette type is 2D and the diskette sector size is 256 bytes. The *STD value is not valid for any other combination of type and sector size.

***BASIC:** The basic exchange type is used. The diskette type must be 1 or 2, and the diskette sector size must be 128 bytes.

***H:** The H exchange type is used. The diskette type must be 2D, and the diskette sector size must be 256 bytes.

***I:** The I exchange type is used. The diskette type and sector size may be any of the following:

Diskette type

Sector size (bytes)

1	128, 256, or 512
2	128, 256, or 512
2D	256, 512, or 1024

FILETYPE parameter

The FILETYPE parameter specifies whether the database file description describes data records or source records. Further, it specifies whether each member of a database file being created is to contain data records or source records (statements). For example, the file could contain RPG source statements for an RPG program or data description source (DDS) statements for another device or database file.

Note: If you are creating a source type *physical* database file and are not providing field-level descriptions of the file (through data description specifications (DDS)), you can use either the Create Physical File (CRTPF) command or the Create Source Physical File (CRTSRCPF) command. However, the CRTSRCPF command is usually more convenient and efficient, because it is designed to be used to create source physical files. If DDS is provided when you are creating a source type database file, you should use the CRTPF command or the Create Logical File (CRTLFL) command, which both have the SRCFILE and SRCMBR parameters for specifying source input.

Records in a source file must have at least three fields: the first two are the source sequence number field and the date field; the third field contains the source statement. These three fields are automatically provided by the OS/400 when a source file is created for which no DDS is provided; additional source fields can be defined in DDS. The length of the sequence number field must be six zoned digits with two decimal places. The length of the date field must be six zoned digits with no decimal places.

The source sequence number and date fields are added to the source record when:

- Records are read into the system.
- Records are created by the Source Entry Utility (which is part of the licensed Application Development* Tools program).

The fields are added when an inline data file (specified as the standard source file format) is read from the diskette device. The spooling reader places a sequence number in the source sequence number field and sets up a zeroed date field.

If those fields already exist in records read from the diskette device, they are not changed. If the records in a database file are in source format and are being read as an inline data file in data format, the source sequence number and date fields are removed.

For more information about data and source files, see Database Programming topic in the Information Center.

Values allowed

***DATA:** The file created contains or describes data records.

***SRC:** The file created contains or describes source records. If the file is keyed, the 6-digit source sequence number field must be used as the key field.

FRCRATIO parameter

The force write ratio (FRCRATIO) parameter specifies the maximum number of records that can be inserted, updated, or deleted before they are forced into auxiliary (permanent) storage. The force write ratio ensures that all inserted, updated, or deleted records are written into auxiliary storage at least as often as this parameter specifies. In the event of system failure, the only records likely to be lost would be those that were inserted, updated, or deleted since the last force write operation.

The force write ratio is applied to all records inserted, updated, or deleted in the file through the open data path (ODP) to which the force write ratio applies. If two programs are sharing the file, SHARE(*YES), the force write ratio is not applied separately to the set of records inserted, updated, or deleted by each program. It is applied to any combination of records (from both programs) that equals the specified force write ratio parameter value. For example, if a force write ratio of 5 was specified for the file, any combination of five records from the two programs (such as four from one program and one from the other) forces the records to be written to auxiliary storage. If two or more programs are using the file through separate ODPs, the insertions, updates, and deletions from each program are accumulated individually for each ODP.

Each database file can have a force write ratio assigned to it. Logical files, which can access data from more than one physical file, can specify a more restrictive force write ratio (a smaller number of records) than that specified for the based-on physical files. However, a logical file cannot specify a *less* restrictive force write ratio. If a logical file specifies a less restrictive force write ratio than that specified for any of the physical files, the most restrictive force write ratio from the physical files is used for the logical file. For example, if the force write ratios of three physical files are 2, 6, and 8, the force write ratio of a logical file based on these physical files cannot be greater than 2. If no force write ratio is specified for the logical file, 2 is assumed. Thus, each time a program inserts, updates, or deletes two records in the logical file (regardless of which physical files are affected), those records are forced into auxiliary storage.

The FRCRATIO number overrides the SEQONLY number specified. For example, if you specify:

```
0VRDBF ... SEQONLY(*YES 20) FRCRATIO(5)
```

The value of 20 is overridden and a buffer of five records is used. When FRCRATIO(1) is used, a buffer still exists, but it contains only a single record.

Access paths associated with the inserted, updated, and deleted records are written to auxiliary storage only when all the records covered by the access path have been written to auxiliary storage. If only one ODP exists for the file, the access path is forced to auxiliary storage whenever a forced write occurs. If two or more ODPs to the file exist, the access path is written to auxiliary storage whenever all the inserted, updated, and deleted records for all the ODPs have been forced.

Notes:

- These rules apply only when a force write ratio of 2 or higher is specified. When a force write ratio of 1 is specified, the access path is not written to auxiliary storage until all the ODPs have been closed.
- If the file is being recorded in a journal. FRCRATIO(*NONE) should be specified. More information is in

the Backup and Recovery  book.

Values allowed

***NONE:** There is no specified ratio; the system determines when the records are written to auxiliary storage.

number-of-records-before-force: Specify the number of updated, inserted, or deleted records that are processed before they are explicitly forced to auxiliary storage.

IGCFEAT parameter

The IGCFEAT parameter specifies which double-byte character set (DBCS) table is used, according to device and language. The following table indicates the corresponding IGCFEAT parameter and DBCS font table for the double-byte character set device being configured.

Table 7. DBCS Features Configurable on the IGCFEAT Parameter

Language/Device	Type of Physical DBCS Work Station	Configure as Type-Model	Configure with DBCS Feature
Japanese Display Stations	5295-001 Display	5555-B01	((2424J4 55FE))
	5295-002 Display	5555-B01	((2424J4 68FE))
	InfoWindow 3477-J Display	5555-B01, C01	((2424J4 68FE))
	PS/55 with 5250PC	5555-B01	((2424J4 68FE))
	PS/55* with graphics 5250PC	5555-G01	((2424J4 68FE))
	PS/55* with graphics 5250PC	5555-G02	((2424J4 68FE))
	PS/55 with 5250PC/2	5555-E01	((2424J0 (1)))
Japanese 24x24 Printers	3270-type Display	3279-0	((2424J0 (1)))
	PS/55 with Client Access/400 by OS/400	5555-B01	((2424J0 (1)))
	Attached to 5295-001 Display	5553-B01	((2424J1 55FE))
	Attached to 5295-002 Display	5553-B01	((2424J1 68FE))
	Attached to PS/55	5553-B01	((2424J1 68FE))
	5227-001 Printer	5553-B01	((2424J2 55FE))
Japanese 32x32 Printers	5327-001 Printer	5553-B01	((2424J2 68FE))
	5337-001 Printer	5553-B01	((3232J0 (1)))
Korean Display Stations	5383-200 Printer	5583-200	((3232J0 (1)))
	5250-Type Display	5555-B01	((2424K0 (1)))
Korean 24x24 Printers	3270-Type Display	3279-0	((2424K0 (1)))
	Attached to 5295 Display	5553-B01	((2424K0 (1)))
	Attached to PS/55	5553-B01	((2424K0 (1)))
Traditional Chinese Display Stations	5227-002 Printer	5553-B01	((2424K2 52FE))
	5250-Type Display	5555-B01	((2424C0))
Traditional Chinese 24x24 Printers	3270-Type Display	3279-0	((2424C0))
	Attached to 5295 Display	5553-B01	((2424C0))
	Attached to PS/55	5553-B01	((2424C0))
Simplified Chinese Display Stations	5227-003 Printer	5553-B01	((2424C2 5CFE))
	5250-Type Display	5555-B01	((2424S0))
Simplified Chinese 24x24 Printers	3270-Type Display	3279-0	((2424S0))
	Attached to PS/55	5553-B01	((2424S0))
	5227-005 Printer	5553-B01	((2424S2 6FFE))

JOB parameter

The JOB parameter specifies the name of the job to which the command is applied. The job name identifies all types of jobs on the system. Each job is identified by a qualified job name, which has the following format:

job-number/user-name/job-name

Note: Although the syntax is similar, job names are qualified differently than OS/400 object names.

The following list describes the pieces of the qualified job name:

- Job number: The job number is a unique 6-digit number that is assigned to each job by the system. The job number provides a unique qualifier if the job name is not otherwise unique. The job number can be determined by means of the Display Job (DSPJOB) command. If specified, the job number must have exactly six digits.
- User name: The user name identifies the user profile under which the job is to run. The user name is the same as the name of the user profile and contains a maximum of 10 alphanumeric characters. The name can come from one of several sources, again, depending on the type of job:
 - Batch job: The user name is specified on the SBMJOB command, or it is specified in the job description referenced by the BCHJOB or SBMJOB commands.
 - Interactive job: The user name is specified at sign-on, or the user name is provided from the default in the job description referred to by the work station's job entry.
 - Autostart job: The user name is specified in the job description referred to by the job entry for the autostart job.
- Job name: The job name can contain a maximum of 10 alphanumeric characters, of which the first character must be alphabetic. The name can come from one of three sources, depending on the type of job:
 - Batch job: The job name is specified on the Batch Job (BCHJOB) or Submit Job (SBMJOB) commands or, if not specified there, the unqualified name of the job description is used.
 - Interactive job: The job name is the same as the name of the device (work station) from which the sign-on was performed.
 - Autostart job: The job name is provided in the autostart job entry in the subsystem description under which the job runs. The job name was specified in the Add Autostart Job Entry (ADDAJE) command.

Commands only require that the simple name be used to identify the job. However, additional qualification must be used if the simple job name is not unique.

Duplicate job names

If a duplicate job name is specified in a command in an *interactive* job, the system displays all of the duplicates of the specified job name to the user in qualified form. The job names are displayed in qualified form along with the user name and job number so that you can further identify the job that is to be specified in a command. You can then enter the correct qualified job name.

If a duplicate job name is used in a command in a *batch* job, the command is not processed. Instead, an error message is written to the job log.

Values allowed

The JOB parameter can have one or more of the following values, depending upon the command:

: The job is the one in which the command is entered; that is, the command with JOB() specified on it.

*JOB: The simple job name is the unqualified name of the job description.

*NONE: No job name is specified as in the Display Log (DSPLOG) command.

job-name: A simple job name is specified.

qualified-job-name: You must specify a qualified job name. If no job qualifier (user name and job number) is given, all of the jobs currently in the system are searched for the job name. If duplicates of the specified name are found, a qualified job name must be specified.

LABEL parameter

The LABEL parameter specifies the data file identifier of the data file (on diskette or tape) used in input and/or output operations. The data file can be in either the exchange format or the save/restore format.

Note: The device file commands are used for diskettes and tapes that are in the exchange format only, *not* for those in the save/restore format; user-defined device files are not used in save/restore operations.

Each data file that is on a diskette or tape has its data file identifier stored in its own file label.

On diskette, all of the data file labels are in one place, in the volume label area of that diskette. In addition to the data file identifier, each label contains other information about the file, such as where the file is stored on the diskette (track and sector) and whether the file continues on another diskette (in the case of a multivolume data file).

On tape, the data file label (or header label) of each data file is stored on the tape just before the data in the file. That is, each file on the tape has its own header label and its own data records together as a unit, and one file follows another. In addition to the data file identifier, each label also contains other information about the file, such as the file sequence number, record and block attributes, and whether it is a multivolume data file.

Generally, the data file identifier is an alphanumeric character string that contains no more than 8 characters. However, the maximum length actually depends on several things: what data format is used for the files, whether the files are on diskette or on tape, and CL commands in which the identifiers are specified. The unused portion of the file identifier field should be left blank.

The first character of the data file identifier must be alphabetic (A through Z, \$, #, or @) and the rest of the characters *should* be alphanumeric (A through Z, 0 through 9, \$, #, _, ., and @). You can use special characters if the identifier is enclosed in apostrophes. However, if the diskette or tape is used on a system other than an iSeries 400, the requirements for specifying identifiers on that system must be considered.

Diskette and tape data file identifiers

For *diskettes* in the exchange format, the data file identifier cannot exceed 8 characters (the same limit as for RPG file names). This limitation applies to the following commands: Create Diskette File (CRTDKTF), Change Diskette File (CHGDKTF), Change Spooled File Attributes (CHGSPLFA), Override Diskette File (OVRDKTF), Delete Diskette Label (DLTDKTLBL), Display Diskette (DSPDKT), and Start Diskette Reader (STRDKTRDR).

For *tapes*, the identifier can have as many as 17 characters. However, if a tape is used on a system other than an iSeries 400, a maximum of 8 characters, or a qualified identifier of no more than 17 characters, should be used. If more than 8 characters are used, the identifier should be qualified and enclosed in apostrophes so that no more than 8 characters occur in either part, and the parts are separated by a period; for example, LABEL('TAXES.JAN1980'). This limitation applies to the following commands: Create Tape File (CRTTAPF), Change Tape File (CHGTAPF), Override Tape File (OVRTAPF), and Display Tape (DSPTAP).

Duplicate data file identifiers are not allowed in the same volume, on either diskette or tape. However, the identifier can be the same as the name of the database file written to diskette or tape if the file name contains no more than 8 characters. The diskette and tape data files contain only data, not file descriptions like those of database files. On diskette, the identifiers ERRORSET and SYSAREA cannot be used; they are reserved for special use.

The data file identifier is put on the volume when the data file is put on the volume. For input/output operations, the identifier can be specified in one of the diskette or tape device file commands, or it can be passed as a parameter when the device file is opened by the high-level language program that uses the file.

Save/Restore format

For *diskettes* in the save/restore format, the data file identifier can have a maximum of 17 characters. If a library name is used to generate the label, the file identifier can have a maximum of 15 characters. The identifier consists of a library name of up to 10 characters followed by a period, a Q, and a 3-digit sequence number; for example, LABEL('PAYLIB.Q014'). The 15-character limit applies to identifiers of save/restore data files displayed by the DSPDKT command.

For *tapes* in the save/restore format, the identifier can have a maximum of 17 characters. If a library name is used to generate the label, the identifier cannot exceed 10 characters. You may specify a label other than a library name.

Values allowed

One of the following values can be specified for the LABEL parameter, depending upon the command.

***ALL:** Labels for all the data file identifiers in the specified diskette or tape volumes are shown on the display.

***NONE:** The data file identifier is not specified. It must be supplied before the device file (and/or database file) is opened to be used in the diskette or tape operation.

***SAME:** The data file identifier already present in the diskette or tape device file does not change.

data-file-identifier: Specify the identifier of the data file (on diskette or tape) used or displayed with the device file description.

***LIB:** The file label is created by the system and the name of the library specified on the LIB parameter is used as the qualifier for the file name.

***SAVLIB:** The file label is created by the system, and the name of the library specified on the SAVLIB parameter is used as the qualifier for the file name.

MAXACT parameter


The maximum activity level (MAXACT) parameter specifies the maximum number of jobs that can be concurrently started and that remain active through a job queue entry, communications entry, routing entry, or work station entry. A job is considered active from the time it starts running until it is completed. This includes time when:

- The job is actually being processed.
- The job is waiting for a response from a work station user.
- The job is started and available for processing but is not actually using the processor. For example, it might have used up its time slice and is waiting for another time slice.
- The job is started but is not available for processing. For example, it could be waiting for a message to arrive on its message queue.

Values allowed

***NOMAX:** There is no maximum number of jobs that can be active at the same time.

maximum-active-jobs: Specify a value that indicates the maximum number of jobs that can be concurrently active through this entry.

See the Work Management  book for a description of activity level controls.

OBJ parameter

The object (OBJ) parameter specifies the names of one or more objects affected by the command in which this parameter is used. All of the objects must be in the library specified in the LIB parameter, the SAVLIB parameter, or the library qualifier in the OBJ parameter, depending upon which command is used.

On some commands, the generic name of a group of objects can be specified. To form a generic name, add an asterisk (*) after the last character in the common group of characters; for example, ABC*. If an * is not included with the name, the system assumes that the name is a complete object name.

Values allowed

Depending on the command, the following types of values can be specified on the OBJ parameter:

- *ALL
- Simple object name
- Qualified object name
- Generic object name
- Qualified generic object name

OBJTYPE parameter

The object type (OBJTYPE) parameter specifies the types of OS/400 objects that can be operated on by the command in which they are specified. The object types that can be specified in the OBJTYPE parameter vary from command to command.

The primary purpose of Table 8 (“Table 8. Object Types Used by Commands Containing the OBJTYPE Parameter” on page 160) is to provide you with a list of the object types that can be operated on by commands containing the OBJTYPE parameter. The Xs indicate that the object type (in the same row as the X) can be specified as a value for the OBJTYPE parameter of the command (in the same column as the X).

The table also provides you with an index to the object-related commands that operate on most objects. The object-related commands allow you to perform general functions on most objects without knowing the special commands related to the specific object type. For example, you could use the CRTDUPOBJ command to create a copy of a file or library instead of the specific commands CPYF (Copy File) or CPYLIB (Copy Library). The table can help you determine whether you can use the general object-related commands to perform a function on a specific object.

The following commands also contain the OBJTYPE parameter but are not included in the table because they operate on only a few object types.

- CHKDLO operates on *DOC and *FLR.
- CPROBJ and DCPOBJ operate on *FILE, *MENU, *MODULE, *PGM, *PNLGRP, and *SRVPGM.
- CRTSQLPKG operates on *PGM and *SRVPGM.
- DSPPGMADP operates on *PGM, *SQLPKG, and *SRVPGM.
- DSPPGMREF operates on *PGM and *SQLPKG.
- RSTCFG operates on *CFGL, *CNL, *COSD, *CTLD, *DEVD, *LIND, *MODD, and *NWID.
- SAVLICPGM operates on *LNG and *PGM.
- SETOBJACC operates on *FILE and *PGM.

The DSPLNK and WRKLNK commands operate on all object types.

The ALCOBJ and DLCOBJ commands also require that an object type value is specified. However, for these commands, the object type value is specified as one of four values (in a list of values) on the required parameter OBJ. The following object types can be specified for the OBJ parameter on the ALCOBJ and DLCOBJ commands: *AUTL, *BNDDIR, *CLD, *CRQD, *CSI, *DEVD, *DTAARA, *DTADCT,

*DTAQ, *FCT, *FILE, *FNTRSC, *FORMDF, *IPXD, *LIB, *MENU, *MODULE, *MSGQ, *NODL, *NTBD, *NWS, *OVL, *PAGDFN, *PAGSEG, *PDG, *PGM, *PNLGRP, *PSFCFG, *QMFORM, *QMQR, *QRYDFN, *SBS, *SCHIDX, *SQLPKG, *SRVPGM, *SSND, *S36, *USRIDX, *USRQ, *USRSPC, and *WSCST.

Note: RTVOBJD and DSPOBJD support the same object types. To see which object types are supported by RTVOBJD, refer to the column labeled DSPOBJD in the following table.

Table 8. Object Types Used by Commands Containing the OBJTYPE Parameter:

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General					Commands					
	C H K O B J	D M P O B J	M O V O B J	R N M O B J	W R K O B J	C H G O B J A U D	D S P O B J A U T	E D T O B J A U T	G R T O B J A U T	R V K O B J A U T	C H G O B J D	D S P O B J D	C H G O B J W N	C R T O B J	W R K O B J C K	C H G O B J P G P	D M P S Y S O B J	P R T D S K I N F	R S T O B J	S A V C H G O B J	S A V O B J
*ALRTBL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*AUTL	x	x		x	x	x	x	x			x	x	x	x	x	x	x				
*BLKSF																x		x			
*BNDDIR	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CFG	x	x			x	x	x	x	x	x	x	x	x		x	x	x	x			
*CHRSF																					
*CHTFMT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CLD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CLS	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CMD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CNL	x	x			x	x	x	x	x	x	x	x	x		x	x	x				
*COSD	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x	x			
*CSI	x	x	x		x	x	x	x	x				x	x	x	x	x	x	x	x	x
*CRG																					
*CRQD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CTLD	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x	x			
*DEVD	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x	x			
*DDIR	x	x			x	x	x							x			x	x			
*DIR		x															x	x			
*DOC	x		x	x	x		x	x	x	x	x	x	x	x	x				x	x	x
*DSTMF	x	x			x	x	x	x	x	x	x	x	x		x	x	x	x			
*DTAARA	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x	x

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General										
	CH K O B J	DM P O B J	MO V O B J	RN M O B J	WR K O B J	CH G O B J A U D	DS P O B J A U T	ED T O B J A U T	GR T O B J A U T	RV K O B J A U T	CH G O B J D	DS P O B J D	CH G O B J O W N	CR T D U P O B J	WR K O B J L C K	CH G O B J P G P	DM P S Y S O B J	PR T D S K I N F	RS T O B J	SA V C H G O B J	SA V O B J
*DTADCT		x				x										x	x				
*DTAQ						x															

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General										
	CH K O B J	DM P O B J	MO V O B J	RN M O B J	WR K O B J	CH G O B J A U D	DS P O B J A U T	ED T O B J A U T	GR T O B J A U T	RV K O B J A U T	CH G O B J D	DS P O B J D	CH G O B J O W N	CR T D U P O B J	WR K O B J L C K	CH G O B J P G P	DM P S Y S O B J	PR T D S K I N F	RS T O B J	SA V C H G O B J	SA V O B J
*EDTD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*EXITRG	x	x	x			x	x	x			x	x				x	x	x	x	x	x
*FCT	x	x		x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*FIFO																					
*FILE	x	x		x	x	x	x	x	x	x	x	x		x	x	x	x	x	x	x	x
*FLR	x	x			x		x				x			x		x	x				
*FNTRSC	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*FNNTBL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x
*FORMDF	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*FTR	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*GSS	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*IGCDCT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			x	x	x
*IGCSRT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*IGCTBL	x	x			x	x	x	x	x	x	x			x	x	x	x	x	x	x	x
*IMGCLG	x	x		x	x	x	x	x		x	x	x		x	x	x	x	x	x	x	x
*IPXD	x	x		x	x	x	x	x	x	x	x	x		x	x	x	x				
*JOB	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*JOBQ	x	x	x	x	x	x	x	x	x	x	x	x	x			x		x	x	x	x
*JOBSCD	x	x			x	x	x	x	x	x	x					x		x	x	x	x

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General										
	C H K O B J	D M P O B J	M O V O B J	R N M O B J	W R K O B J	C H G O B J A U D	D S P O B J A U T	E D T O B J A U T	G R T O B J A U T	R V K O B J A U T	C H G O B J D	D S P O B J D	C H G O B J O W N	C R T D U P O B J	W R K O B J L C K	C H G O B J P G P	D M P S Y S O B J	P R T D S K I N F	R S T O B J	S A V C H G O B J	S A V O B J
*JRN	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*JRNRCV	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*LIB	X	X			X	X	X	X	X	X	X	X		X	X	X	X				
*LIND	X	X			X	X	X	X	X	X	X	X		X	X	X	X				
*MEDDFN	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	X
*MENU	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*MGTCOL	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*MODD	X	X			X	X	X	X	X	X	X	X		X	X	X	X				
*MODULE	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*MSGF	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General										
	C H K O B J	D M P O B J	M O V O B J	R N M O B J	W R K O B J	C H G O B J A U D	D S P O B J A U T	E D T O B J A U T	G R T O B J A U T	R V K O B J A U T	C H G O B J D	D S P O B J D	C H G O B J O W N	C R T D U P O B J	W R K O B J L C K	C H G O B J P G P	D M P S Y S O B J	P R T D S K I N F	R S T O B J	S A V C H G O B J	S A V O B J
*MSGQ	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*NODGRP	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*NODL	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X	X	X
*NTBD	X	X		X	X	X	X	X	X	X	X	X		X	X	X	X				
*NWID	X	X		X	X	X	X	X	X	X	X	X		X	X	X	X				
*NWSD	X	X			X	X	X	X	X	X	X	X		X	X	X	X				
*OUTQ	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*OVL	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*PAGDFN	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
*PAGSEG	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General										
	CH K O B J	D M P O B J	M O V O B J	R N M O B J	W R K O B J	C H G O B J A U D	D S P O B J A U T	E D T O B J A U T	G R T O B J A U T	R V K O B J A U T	C H G O B J D	D S P O B J D	C H G O B J O W N	C R T D U P O B J	W R K O B J L C K	C H G O B J P G P	D M P S Y S O B J	P R T D S K I N F	R S T O B J	S A V C H G O B J	S A V O B J
*PDG	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PGM	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PNLGRP	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PRDAVL		x		x		x					x	x	x	x	x	x	x		x		x
*PRDDFN	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
*PRDL0D		x	x	x	x	x					x	x	x	x	x	x	x	x	x		
*PSFCFG	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*QMFORM	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*QMQRV	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*QRYDFN	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*RCT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*SBSD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*SCHIDX	x	x	x	x	x	x	x	x	x	x		x			x	x	x	x	x	x	
*SOCKET															x			x			
*SPADCT	x	x	x	x	x	x	x	x	x	x	x	x	x			x	x	x	x	x	x

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General										
	CH K O B J	D M P O B J	M O V O B J	R N M O B J	W R K O B J	C H G O B J A U D	D S P O B J A U T	E D T O B J A U T	G R T O B J A U T	R V K O B J A U T	C H G O B J D	D S P O B J D	C H G O B J O W N	C R T D U P O B J	W R K O B J L C K	C H G O B J P G P	D M P S Y S O B J	P R T D S K I N F	R S T O B J	S A V C H G O B J	S A V O B J
*SQLPKG	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*SQLUDT		x		x										x							
*SRVPGM	x		x		x	x	x	x	x	x	x	x	x		x	x	x	x	x	x	x
*SSND	x	x			x	x	x	x	x	x	x	x	x		x	x	x	x	x	x	x
*STMF																		x			

Value	Basic Object-Related Commands															Other Object-Related Commands					
	Object					Object Authority					General										
	C H K O B J	D M P O B J	M O V O B J	R N M O B J	W R K O B J	C H G O B J A U D	D S P O B J A U T	E D T O B J A U T	G R T O B J A U T	R V K O B J A U T	C H G O B J D	D S P O B J D	C H G O B J O W N	C R T D U P O B J	W R K O B J L C K	C H G O B J P G P	D M P S Y S O B J	P R T D S K I N F	R S T O B J	S A V C H G O B J	S A V O B J
*SVRSTG		x			x	x	x	x	x	x	x				x	x	x	x	x	x	x
*SYMLNK				x									x				x				
*S36	x	x	x		x	x	x	x	x	x	x	x		x	x	x	x	x	x	x	x
*TBL	x	x			x	x	x	x	x	x	x	x		x	x	x	x	x	x	x	x
*USRIDX	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*USRPRF	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x				
*USRQ	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x	x	x
*USRSPC	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*VLDL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*WSCST	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x	x

Values allowed

***ALL:** All the object types that are allowed in the command, specified by name, and are in the specified library are operated on by the command in which they are specified. *ALL refers only to the object types that apply to that command; refer to the individual command descriptions of the OBJTYPE parameter to see which of the OS/400 object types can be specified.

object-type: Specify the predefined values for the types of objects that are to be operated on by the command.

OUTPUT parameter

You use the OUTPUT parameter to specify whether the output from the display command will be shown on the display, printed, or written to an output file. Basically, the same information is provided in either form; only the format is changed as necessary to present the information in the best format for the device. For example, because there are more lines on a printed page than on a display, column headings are not repeated as often in printed output.

If the output is to be shown on the display, it will be sent to the work station that issued the display command. It will be shown in the format specified in the display device file used by that display command. A different device file is used for the output of each display command, and the file is different for displayed, printed, or written file output. In most cases, the name of the command is part of the file names of either type of device file.

If the output will be printed, it is spooled and an entry is placed on the job's output queue. The output can be printed depending on which device is specified in the Start Printer Writer (STRPRTWTR) command.

Note: Although the IBM-supplied printer files are shipped with SPOOL(*YES) specified, they can be changed to SPOOL(*NO) by the Override with Printer File (OVRPRTF) and Change Printer File (CHGPRTF) commands.

If the OUTPUT parameter is not specified in the display command, the default value * is assumed. The output resulting from this value depends on the type of job that entered the command. The following table shows how the output is produced for interactive and batch jobs.

Output	Interactive Job	Batch Job
*	Displayed	Printed
*PRINT	Printed	Printed

Values allowed

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

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

***OUTFILE:** The only output is to be written to a specified database file.

PRTTXT parameter

The print text (PRTTXT) parameter specifies the text that appears at the bottom of listings and on separator pages. Print text is copied from the job attribute when the job enters the system. Print files that originate on another system do not use the print text on the target system. Print text exists as a job attribute (PRTTXT) for defining the print text of a specific job, and as a system value (QPRTTXT) for the default of jobs with *SYSVAL specified. QPRTTXT is the system-wide default for all jobs.

The print text can be up to 30 characters in length. The text should be centered in the form's width and printed in the overflow area. You should center the desired text within the 30 character field.

If the print text is not blank, the system prints 30 characters of text on the bottom of each page. This text normally follows the overflow line and is preceded by a blank line (if the form's length permits). If the user prints past the overflow line, the print text follows the last line of the user text, again preceded by a blank line when possible. If the overflow line is the last line of the form, the print text also prints on the last line of the form, which may result in the typing over of user text.

The print text for job and file separators is put on the first line of the separator page. A job separator contains print text of the job that created the separator at the time the file was printed. A file separator contains the same print text as the spooled file it precedes.

The print text can be specified for all job types. System and subsystem monitor jobs use the system value. Reader and writer jobs use the system value unless print text is changed in the QSPLxxxx job description associated with the reader or writer.

The print text is determined from several places by using the following hierarchical order. If print text is not specified in one place, the next place in the order is used.

The hierarchical order, beginning with the highest priority, is as follows:

- Override print file value
- Print file value
- Job attribute changed by the Change Job (CHGJOB) command
- Job attribute set by the Submit Job (SBMJOB) or Batch Job (BCHJOB) command

- Job description
- System value

Values allowed

For the system value QPRTTXX, any character string can be specified, with the exception of *SYSVAL. If *BLANK is specified, there will be no print text. For PRTTXX, some of the following values can be selected, depending on the command:

***SAME:** The print text does not change.

***CURRENT:** The print text is taken from the submitting job.

***JOB:** The print text is taken from the job description under which the job is run.

***SYSVAL:** The print text is taken from the system value QPRTTXX.

***BLANK:** There is no text or blanks printed.

'print-text': Specify 30 characters of text. If there are blanks in the text, then apostrophes must be used around the entry. The text should be centered within the field for the text to be centered on the page.

REPLACE parameter

The replace (REPLACE) parameter is used on create commands. It specifies that the existing object, if one exists, is replaced by the object of the same name, library, and object type that is being created. The user of the new object is granted the same authority as for the object being replaced. If the object being replaced is secured by an authorization list, then the new object is secured by the same authorization list. The public authority of the new object is the same as the public authority of the replaced object. The AUT parameter from the create command is ignored. All private authorities from the replaced object are copied to the new object. The owner of the new object is *not* copied from the replaced object. The owner of the new object is the creator of the new object or the creator's group profile. Some objects such as panel groups, display files, and menus cannot be replaced if they are in use by the current job or another job.

If the object being created is a program or service program, then the user profile (USRPRF parameter) value from the replaced program is used. The user profile (USRPRF parameter) value from the Create Program or Create Service Program command is ignored. If the value of the user profile (USRPRF parameter) of the program or service program being replaced is *OWNER, then only the current owner of the program or service program being replaced can create the new program or service program that replaces the existing program or service program. If the owner of the existing object and the object being created do not match, the object is not created and message CPF2146 is sent.

If the object being created is a program or service program, then the use adopted authority (USEADPAUT) value from the replaced program or service program is used as long as the user creating the object can create programs/service programs with the USEADPAUT(*YES) attribute. The QUSEADPAUT system value determines whether or not users can create programs or service programs to use adopted authority. For example, if the existing object being replaced has USEADPAUT(*YES) and you do not have authority to create a program or service program that uses adopted authority, the program or service program created will have USEADPAUT(*NO). In this case, the USEADPAUT value was not copied. If you have authority to create programs or service programs that use adopted authority, the created program or service program will have the same USEADPAUT value as the program or service program being replaced. An informational message is sent which indicates whether the USEADPAUT value was copied to the object being replaced.

If the object being created is a file, and the default, or *YES, is specified on the REPLACE parameter, an existing device file other than save file and a DDM file with the same qualified name will be replaced by the new file. For example, an existing display file can be replaced by a new printer file, or tape file, etc.

Object management (*OBJMGT), object existence (*OBJEXIST), and read (*READ) authorities are required for the existing object to allow replacement of the existing object with a new object.

| The existing object is renamed and moved to library QRPLOBJ or library QRPLxxxx if the object resides
| on an Independent ASP (where 'xxxx' is the number of the primary ASP of the ASP group) when the
| creation of the new object is successful. The replaced object is renamed with a Q appended to a time
| stamp and moved to library QRPLOBJ or library QRPLxxxx if the object resides on an Independent ASP.

Restriction: Programs can be replaced while they are being run; however, if the replaced program refers to the program message queue after the renaming of the replaced program to the Qtimestamp name, the program fails and an error message is sent stating that the program message queue is not found.

A database file, physical or logical, and a save file cannot be replaced by any file.

| Library QRPLOBJ is cleared when an initial program load (IPL) of the system is done. Library QRPLxxxx
| is cleared when the primary ASP of the ASP group is varied on.

Values allowed

***YES:** The system replaces the existing object with the new object being created that has the same name, library, and object type.

***NO:** The system does not replace the existing object that has the same name, library, and object type with the object being created.

Scheduling priority parameters (JOBPTY, OUTPTY, PTYLMT)

The scheduling priority parameters specify the priority values used by the system to determine the order in which the jobs and spooled files are selected for processing. Each job is given a scheduling priority that is used for both job selection and spooled file output. The job scheduling priority is specified by the JOBPTY parameter in commands like the Batch Job (BCHJOB), Submit Job (SBMJOB), Create Job Description (CRTJOB), and Change Job Description (CHGJOB) commands. The priority for producing the spooled output from a job is specified by the OUTPTY parameter in the same commands.

In addition, because every job is processed under a specific user profile, the priority for jobs can be limited by the PTYLMT parameter specified on the Create User Profile (CRTUSRPRF) and Change User Profile (CHGUSRPRF) commands. This parameter value controls the maximum job scheduling priority and output priority that *any* job running under a user profile can have; that is, the priority specified in the JOBPTY and OUTPTY parameters of any job command cannot exceed the priority specified in the PTYLMT parameter for that user profile. The scheduling priority is used to determine the order in which jobs are selected for processing and is not related to the process priority specified in the class object.

The three scheduling priority parameters specify the following:

- The PTYLMT parameter specifies the *highest* scheduling priority for *any* job that you submit. In the commands that affect the user profile, the PTYLMT parameter specifies the highest priority that can be specified in another JOBPTY parameter on commands relating to each specific job. You can specify a lower priority for a job on the command used to submit the job. If you specify a higher priority for JOBPTY in the BCHJOB or SBMJOB command than is specified for PTYLMT in the associated user profile, an error message is shown on the display and the maximum priority specified in PTYLMT is assumed. If a higher job priority is specified in the CHGJOB or CHGJOB command, an error message is shown and the attributes are not changed.
- The JOBPTY parameter specifies the priority value to be used for a *specific* job being submitted. In the commands relating to a specific job being submitted, the JOBPTY parameter specifies the actual scheduling priority for the job.

- The OUTPTY parameter specifies the priority for producing the output from all spooled output files from the job. The priority value specified in the OUTPTY parameter determines the order in which spooled files are handled for output. The same value is applied to all the spooled files produced by the job.

The scheduling priority can have a value ranging from 0 through 9, where 1 is the highest priority and 9 is the lowest priority. Any job with a priority of 0 is scheduled for processing before all other jobs that are waiting and that have priorities of 1 through 9.

The priority parameters can be specified on the following commands.

JOBPTY	OUTPTY	PTYLMT
ADDJOBJS	ADDJOBJS	
BCHJOB	BCHJOB	CHGUSRPRF
CHGJOB	CHGDKTF	CRTUSRPRF
CHGJOBBD	CHGJOBBD	
CHGJOBJS	CHGJOBJS	
CRTJOBBD	CHGJOB	RTVUSRPRF
SBMJOB	CHGJOBBD	
SBMJOBJS	CHGJOBJS	
	CHGPJ	
	CHGPRTF	
	CHGSPLFA	
	CRTDKTF	
	CRTJOBBD	
	CRTPRTF	
	OVRDKTF	
	OVRPRTF	
	SBMJOB	
	SBMJOBJS	

Values allowed

Depending upon the command, one or more of the following values apply to the parameter.

5: If a value is not specified in the CRTUSRPRF command, five is the default value that is assumed for the priority limit for the user profile. That would be the highest priority that the user could specify for any job he submits for processing. If not specified in the CRTJOBBD command, five is the default value for both the job scheduling priority and the output priority.

***SAME:** The priority assigned, or the highest priority that can be assigned, does not change.

***JOBBD:** The scheduling priority for the job is obtained from the job description under which the job runs.

scheduling-priority: Specify a priority value ranging from 0 through 9, where 0 is the highest priority and 9 is the lowest priority. Priority 0 is allowed only on CHGJOB.

SEV parameter

The severity (SEV) parameter specifies the severity code that:

- Describes the level of severity associated with an error message.
- Indicates the minimum severity level that causes a message to be returned to a user or program.
- Causes a batch job to end.
- Causes processing of a command to end if a syntax error of sufficient severity occurs.

Note: The LOG parameter on some commands also uses these severity codes for logging purposes (to control which job activity messages and error messages are logged in the job log).

The severity code is a 2-digit number that can range from 00 through 99. The higher the value, the more severe or important the condition. The severity code of a message that is sent to a user indicates the

severity of the condition described by the message. More than one message can have the same severity code. If a severity code is not specified for a predefined message, it is assumed to be 00 (information only).

You can specify a severity code for any message when it is defined by the Add Message Description (ADDMSGD) command. To change the severity code of a message, use the Change Message Description (CHGMSGD) command.

IBM-defined severity codes are used in all of the IBM-supplied messages that are shipped with the system.

00 - Information: A message of this severity is for information purposes only; no error was detected and no reply is needed. The message could indicate that a function is in progress or that it has reached a successful completion.

10 - Warning: A message of this severity indicates a potential error condition. The program may have taken a default, such as supplying missing input. The results of the operation are assumed to be what was intended.

20 - Error: An error has been detected, but it is one for which automatic recovery procedures probably were applied, and processing has continued. A default may have been taken to replace input that was in error. The results of the operation may not be valid. The function may be only partially complete; for example, some items in a list may be processed correctly while others may fail.

30 - Severe Error: The error detected is too severe for automatic recovery, and no defaults are possible. If the error was in source data, the entire input record was skipped. If the error occurred during program processing, it leads to an abnormal end of the program (severity 40). The results of the operation are not valid.

40 - Abnormal End of Program or Function: The operation has ended, possibly because it was unable to handle invalid data, or possibly because the user ended it.

50 - Abnormal End of Job: The job was ended or was not started. A routing step may have ended abnormally or failed to start, a job-level function may not have been performed as required, or the job may have been ended.

60 - System Status: A message of this severity is issued only to the system operator. It gives either the status of or a warning about a device, a subsystem, or the whole system.

70 - Device Integrity: A message of this severity is issued only to the system operator. It indicates that a device is malfunctioning or in some way is no longer operational. You may be able to restore system operation, or the assistance of a service representative may be required.

80 - System Alert: A message of this severity is issued only to the system operator. It warns of a condition that, although not severe enough to stop the system now, could become more severe unless preventive measures are taken.

90 - System Integrity: A message of this severity is issued only to the system operator. It describes a condition that renders either a subsystem or the whole system inoperative.

99 - Action: A message of this severity indicates that some manual action is required, such as specifying a reply, changing printer forms, or replacing diskettes.

SPLNBR parameter

The spooled file number (SPLNBR) parameter is used when more than one spooled file is created by a job and the files all have the same name. The files are numbered, starting with 1, in the order that they are opened by the job. The job log is always the last file for a job.

A file number is generated for each file when it is opened within a job (when output records are produced) and it is used by the system as long as the job and/or the files are on the system. If the files are not uniquely named because they were opened more than once, this file number is used to specify which file (or group of records, if the complete file has not yet been produced) is acted upon by a CL command.

TEXT parameter

The TEXT parameter specifies the user-defined description that briefly describes the object being created or changed. The description can include up to 50 characters; if it is a quoted string (that is, enclosed in apostrophes), any of the 256 EBCDIC characters can be used. The apostrophes are not required if the string does not contain any blanks or other special characters. Any of the 50 character positions not filled by the specified description are padded with blanks.

The description is used to describe any of the OS/400 objects when the named object is shown on the display by the Display Object Description (DSPOBJD) command. Only objects for which object operational authority has been obtained can be displayed by a user. See the OBJTYPE parameter description for a list of the OS/400 object types.

For commands that use a database source file to create some type of object, you can (by default) use the text from the source file member as the text for the newly-created object. For example, if you use the Create Control Language Program (CRTCLPGM) command to create a CL program, but you do not specify a description in the TEXT parameter, the text specified for the source file member (SRCMBR parameter) of the source file (SRCFILE parameter) is assumed as the descriptive text for the CL program.

Values allowed

Depending upon the command, one or more of the following values apply to the TEXT parameter.

***SRCMBRTXT:** For commands that create objects based on database source files only, the text is taken from the source member. If a device or an inline file is used for source input or if source is not used, the text is left blank.

***BLANK:** The user description of the object being created or changed is left blank.

***SAME:** The user-defined description does not change.

'description': Specify the description of the object being created or changed. Up to 50 characters enclosed in apostrophes (required for blanks and other special characters) can be specified to describe the object. If an apostrophe is one of the 50 characters, two apostrophes (") must be used instead of one to represent the apostrophe character.


VOL parameter

The volume (VOL) parameter specifies the volume identifiers of the volumes used in a diskette, tape, or optical operation. A diskette volume consists of a single diskette. A tape volume consists of a reel of tape. An optical volume consists of a single side of an optical cartridge or a single CD-ROM. Optical cartridges are dual sided and each side is a separate volume.

The volume identifier is the identifier stored on each diskette, tape, or optical disk (in the volume label area) that it identifies. The diskettes (volumes) must be on the diskette drive in the same order as the identifiers are specified in the VOL parameter. An inquiry message is sent to the system operator if a volume identifier is missing or out of order.

Tape volumes must be on the tape units in the same order as their identifiers are specified in the VOL parameter and as the device names are specified in the DEV parameter of the tape device file commands. However, if the tapes are read backward (a function supported in COBOL), the volumes must be in reverse order to that specified in the VOL parameter. Nevertheless, the device names are still specified in forward order in the DEV parameter.

In general, the rule for specifying diskette and tape volume identifiers is that as many as 6 characters, containing any combination of letters and digits, can be used. Special characters can be used if the identifier is enclosed in apostrophes. However, if the diskette or tape is used on a system other than an iSeries 400, the requirements for specifying identifiers on that system must be considered.

Optical volume identifiers can be up to 32 characters long and can contain any combination of digits and upper case letters. Each optical volume identifier must be unique. No two optical volumes with the same identifier can be present on the system at the same time. A complete list of the rules for optical volume identifiers can be found in the Optical Support  book.

For diskettes in the data exchange format and for labeled tapes, the following rules apply:

- **Characters:** A maximum of 6 characters, or fewer, can be specified for each volume identifier. Alphabetic and numeric characters can be used in any order.
- **Uniqueness:** More than one volume can have the same identifier. You may have a file using the same identifier for several volumes; in this case, the system keeps track of the order internally with a sequence number written on the volumes. However, volume identifiers should be unique whenever possible.
- **Order:** When multiple volumes (with different identifiers) are used in a single operation, they must be in the same order as the volume identifiers specified in the VOL parameter.

Multivolume files

For a multivolume file on diskettes (that is, a data file on several diskettes, all having the same name), a message is sent to the system operator for each diskette volume after the first, until all volumes have been processed. If (for S/R only) more than 100 diskettes are used for the same file, duplicate diskette sequence numbers occur for each additional diskette used after the first hundred (01 through 99, and 00). For each 100 diskettes written for the file, a message is sent to the system operator indicating the total number written. When the diskettes are read, the operator must determine the order in which the diskette volumes are inserted.

If multiple volumes (tapes or diskettes) are used in an operation and all have the same volume identifier, that identifier must be specified in the VOL parameter once for each volume used. For example, if three tapes named QGPL are used in a save operation, VOL(QGPL QGPL QGPL) must be specified.

When a multivolume file on *tape* is processed and multiple tape units are used, the tape volumes must be placed in the tape devices in the same order as they are specified in the VOL parameter. For example, if five volumes and three tape units are used, they are mounted as follows: VOL1 on unit 1, VOL2 on unit 2, VOL3 on unit 3, VOL4 on unit 1, and VOL5 on unit 2.

Values allowed

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

***NONE:** No volume identifier is specified.

***SAME:** Previously specified volume identification does not change.

***SAVVOL:** The system, using the save/restore history information, determines which tape or diskette volumes contain the most recently saved version. If the device specified in the DEV parameter of the restore command does not match the device of the most recently saved version of the object, an error message is returned to the user, and the function is ended. If the wrong volume is mounted in the unit specified by the command, a message is returned to the system operator that identifies the first volume that must be placed in the device before the restore operation can begin.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are put on the device and used. Each tape or diskette volume identifier contains a maximum of 6 alphanumeric characters. Each optical volume identifier contains a maximum of 32 characters. A blank is used as a separator character when listing multiple identifiers.

WAITFILE parameter

You use the WAITFILE parameter to specify the following:

- For the maximum number of seconds that a program waits for file resources to be allocated when the file is opened
- For session resources when the evoke function is issued for an APPC device
- For the device to be allocated when an acquire operation is performed to read the file

If the program must wait, it will be placed in a wait state until the resources are available or until the wait time expires. If two or more file resources are needed and are not available because they are being used by different system users, the acquisition of each resource might require a wait. This maximum is applied to each wait.

The length of the wait can be specified in this parameter, or the default wait time of the class that applies to the object can be used. If the file resources cannot be allocated in the specified number of seconds, an error message is returned to the program.

The file resources that must be allocated depend on the type of file being opened. File resources consist of the following.

- For device files that are not spooled (SPOOL(*NO)), the file resources include the file description and device description. Because the device description must be allocated, the device itself must also be available.
- For device files that are spooled (SPOOL(*YES)), the file resources include the file description, the specified output queue, and storage in the system for the spooled data. Because the data is spooled, the device description (and thus the device itself) need not be available.
- For database files, the file resources consist of the file and member data. The file's associated member paths are not accessed, and therefore, the system does not wait for them. A file open exception error can occur before the WAITFILE time has expired when an access path is not available (for example, when the access path is being rebuilt).

The Allocate Object (ALCOBJ) command can be used to allocate specific file resources before the file is opened.

The session resources that were allocated for an APPC device conversation can be lost between the time the application issues a detach function or receives a detach indication and the time another evoke function is issued. If the session resource is lost, this parameter is used to determine the length of time that the system waits for another session resource.

Values allowed

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The default wait time specified in the class description is used as the wait time for the file resources to be allocated.

number-of-seconds: Specify the maximum number of seconds that the program waits for the file resources to be allocated. Valid values range from 1 through 32767 seconds.

Database and device files used by CL commands

Many of the IBM-supplied CL commands use database and device files during processing. This section provides a cross-reference between the commands and the IBM-supplied files used by the commands. All of the commands and files for *all* iSeries 400 licensed programs that meet the following criteria are included:

- The types of files included are:
 - Database files: physical (PF) and logical (LF), including files with data and files used as model files (no data)
 - Device files: diskette (DKTF), tape (TAPF), and printer (PRTF)
- The files are included only if a user might have a reason to use them, such as declaring them in a program so they can be overridden with another file. Two examples:
 - You want to change some attributes for a printer device file, such as the font to be used or the printed lines per inch (the FONT or LPI parameter).
 - You want to override the IBM-supplied file with your own output file (when valid).

The types of files **not** included in this section are:

- iSeries 400-provided display (DSPF) device files, because they should not be changed or overridden.
- Most of the iSeries 400-provided database files used by *directory* commands, *document library object* commands, and *optical index database* files because they cannot be overridden.

As mentioned, IBM-supplied physical (PF) or logical (LF) files that are used as **model files** for certain commands are included in the following table. (Examples are the model files listed under the DSPFD, DSPJRN, and STRPFRMON commands.) In most cases, these model files do not contain data; instead, they contain the *definitions* (or record formats) of the files to be created for storing the actual output data resulting from use of these commands. For the record formats of these files, you can display the file's description online or you can refer to the publication or soft-copy manual that documents that command.

Additionally, some files considered to be IBM-supplied files do not actually exist on the system until some function is used that requires the file and so creates it at that time.

The following notes describe how the table of commands and files are sorted and explain the meanings of the superscripts used in the table.

Notes for Table:

1. The first column of the table lists the CL commands that use the iSeries 400-provided files shown in the third column. The table entries are in alphabetic order by *command name* first. If there is more than one library for a command, they are further ordered by the file *library name* and then by *file name* within each library.
2. **A superscript 1 (¹)** following the description of a file indicates that the file is used only when the output from the command is directed to that *form* of output-by an output-related parameter on the command.
 - The superscript ¹ is used at the end of *printer file* (PRTF) descriptions to show that use of the printer file is dependent on the job environment and the print-related value specified (or assumed) on the command. For commands with an OUTPUT parameter (primarily DSPxxx and WRKxxx commands), the output is printed either when OUTPUT(*) is specified in a batch job, or when OUTPUT(*PRINT) is specified in a batch or interactive job. For other commands, the output is printed if an *PRINT, *LIST, and/or *SRC value is specified on a different parameter.
 - The superscript ¹ is used at the end of *database file* (PF or LF) descriptions to show that use of the database file is dependent on the print-related value specified (or assumed) on the command. For commands with an OUTPUT parameter (primarily DSPxxx and SAVxxx commands), the output is directed to the database file when OUTPUT(*OUTFILE) is specified. The same is true for other commands that specify a value similar to *OUTFILE on one of its parameters.
3. **A superscript 2 (²)** following a file's description indicates that the file is a *model file* and not an output file. As a model file, it defines the record format of the file created to contain the actual output.
4. Those files showing "user-lib" as the file library do not exist on the system until the user creates them. When the command is used, the file is created in the user's specified library with the file name shown.

Table 2. Files Used by CL Commands

Command Name	File Library	File Name	File Type	File Usage
ADDDOCCVN	QUSRSYS	QAO1CRL	LF	Document conversion <i>logical</i> file for input or output.
	QUSRSYS	QAO1CVNP	PF	Document conversion <i>physical</i> file for input or output.
ADDSTQ	QUSRSYS	QAO1DCVN	PRTF	Document conversion printer file.
	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.
	QUSRSYS	QASNADSR	PF	SNADS routing table.
ADDSTSYSN	QUSRSYS	QASNADSA	PF	SNADS secondary node ID table.
ADDNETJOB	QUSRSYS	QANFNJE	PF	Network job entry database file.
ADDSCOCE	QUSRSYS	QAALSOC	PF	Sphere of control file.
ADDTAPCTG	QUSRSYS	QATAMID	PF	Cartridge ID db file.
	QUSRSYS	QLTAMID	LF	Cartridge ID logical file.
	QUSRSYS	QATACGY	PF	Category db file.
	QUSRSYS	QLTACGY	LF	Category logical file.
ADDTCPHTE	QUSRSYS	QATOCHOST	PF	TCP/IP host file.
ADDTCPIFC	QUSRSYS	QATOCIFC	PF	TCP/IP configuration file.
ADDTCPPORT	QUSRSYS	QATOCPORT	PF	TCP/IP configuration file.
ADDTCPRSI	QUSRSYS	QATOCRSI	PF	TCP/IP configuration file.
ADDTCPRTE	QUSRSYS	QATOCRTE	PF	TCP/IP configuration file.
ADDTXTIDXE	QUSRSYS	QABBADMTB	PF	OfficeVision text search services administration table.
ANSQST	QSYS	QPQAPRT	PRTF	Q & A printer file.
ANZACCGRP	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QAPTPAGD	PF	Performance database input file of files and programs in a process access group (PAG).

Command Name	File Library	File Name	File Type	File Usage
	QPFR	QPPTPAG	PRTF	Printer file of PAG data containing environment, job, file, and program summary data.
ANZDBF	QPFR	QAPTAZDR	PF	Performance data collection file: analyze application database files data.
	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPPTANZD	PRTF	Performance printer file showing physical-to-logical and logical-to-physical database file relationships.
ANZDBFKEY	QPFR	QAPTAZDR	PF	Performance input file of analyze application database files data showing logical file key structures.
	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPPTANKM	PRTF	Performance printer file containing logical file key structure data.
	QPFR	QPPTANZK	PRTF	Performance printer file containing access path and record selection data.
ANZPFRDTA	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = ASYN, BSC, CIOP, CONF, DIOP, DISK, ECL, ETH, HDLC, IDLC, JOBS, LAPD, LIOP, MIOP, POOL, SYS, and X25. ²
	QPFR	QPAVPRT	PRTF	Performance printer file containing the advisor report. ¹
ANZPGM	QPFR	QAPTAZPD	PF	Performance data collection file: analyze application programs data.
	QPFR	QPPTANZP	PRTF	Performance printer file showing program-to-file and file-to-program relationships.
ASKQST	QSYS	QPQAPRT	PRTF	Q & A printer file.
CFGDSTSRV	QUSRSYS	QASNADSA	PF	SNADS secondary node ID table.
	QUSRSYS	QASNADSQ	PF	SNADS destination queues table.
	QUSRSYS	QASNADSR	PF	SNADS destination systems routing table.
CFGTCP	QUSRSYS	QATOCIFC	PF	TCP/IP interface file.
	QUSRSYS	QATOCRTE	PF	TCP/IP route file.
	QUSRSYS	QATOCTCPIP	PF	TCP/IP attributes file.
	QUSRSYS	QATOCPORT	PF	TCP/IP port restrictions file.
	QUSRSYS	QATOCRSI	PF	TCP/IP RSI file.
	QUSRSYS	QATOCHOST	PF	TCP/IP hostname file.
	QUSRSYS	QATOCPS	PF	TCP/IP services file.
CFGTCPSMTP	QUSRSYS	QATMSMTP	PF	TCP/IP SMTP file.
	QUSRSYS	QATMSMTPA	PF	TCP/IP SMTP file.
CHGDOCCVN	QUSRSYS	QAO1CRL	LF	Document conversion <i>logical</i> file for input or output.
CHGDTA	QIDU	QDTALOG	PRTF	Audit control log printer file.
	QIDU	QDTAPRT	PRTF	Record and accumulator total printer file.
	QSYS	QPDZDTALOG	PRTF	DFU run-time audit log.
	QSYS	QPDZDTAPRT	PRTF	DFU run-time printer data file.
CHGDSTQ	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.

Command Name	File Library	File Name	File Type	File Usage
CHGDSTRTE	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.
	QUSRSYS	QASNADSR	PF	SNADS routing table.
CHGFTPA	QUSRSYS	QATMFTP	PF	TCP/IP FTP configuration file.
CHGHTTPA	QUSRSYS	QATMHTTP	PF	TCP/IP HTTP file.
CHGLPDA	QUSRSYS	QATMLPD	PF	TCP/IP LPD configuration file.
CHGPOPA	QUSRSYS	QATMPOPA	PF	POP server configuration file.
CHGPRB	QUSRSYS	QASXNOTE	PF	Problem log user notes file.
	QUSRSYS	QASXPROB	PF	Problem log problem file.
CHGQSTDB	QSYS	QPQAPRT	PRTF	Q & A printer file.
CHGSMTPA	QUSRSYS	QATMSMTP	PF	TCP/IP SMTP configuration file.
CHGTAPCTG	QUSRSYS	QATAMID	PF	Cartridge ID database file.
	QUSRSYS	QATAMID	LF	Cartridge ID logical database file.
CHGTCPA	QUSRSYS	QATOCTCPIP	PF	TCP/IP attributes file.
CHGTCPHTE	QUSRSYS	QATOCHOST	PF	TCP/IP host file.
CHGTCPIFC	QUSRSYS	QATOCIFC	PF	TCP/IP interface file.
CHGTCPRTE	QUSRSYS	QATOCRTE	PF	TCP/IP routes file.
CHGTELNA	QUSRSYS	QATMTELN	PF	TCP/IP TELNET configuration file.
CHKTAP	QSYS	QSYSTAPE	TAPF	Printer file for tape output.
CLRDKT	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
CMPJRNIMG	QSYS	QPCMPIMG	PRTF	Compare journal images printer file.
CPYF	QSYS	QSYSPRT	PRTF	Copy file printer file. ¹
CPYFRMDKT	QSYS	QSYSPRT	PRTF	Copy file printer file. ¹
CPYFRMQRYF	QSYS	QSYSPRT	PRTF	Copy file printer file. ¹
CPYFRMTAP	QSYS	QSYSPRT	PRTF	Copy file printer file. ¹
CPYIGCTBL	QSYS	QSYSDKT	DKTF	Diskette device file used for input and output.
	QSYS	QSYSTAP	TAPF	Tape device file used for input and output.
CPYPFRTDA	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = ASYN, BSC, BUS, CIOP, CONF, DIOP, DISK, DMPT, ECL, ETH, HDLC, IOBS, JOBS, LIOP, MIOP, POOL, RESP, RWS, SBS, SYS, TSK, and X25. ²
CPYPTF	QSYS	QSYSDKT	DKTF	Diskette device file used for input and output.
	QSYS	QSYSTAP	TAPF	Tape device file used for input and output.
CPYSRCF	QSYS	QSYSPRT	PRTF	Copy file printer file. ¹
CRTAPAR	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
CRTBND	QGPL	QCSRC	PF	C/400 source default input file.
CRTBND	QGPL	QCBLLSRC	PF	ILE COBOL/400 source default input file.
	QSYS	QSYSPRT	PF	ILE COBOL/400 source listing printer file.
CRTBNDCL	QGPL	QCLSRC	PF	CL source default input file.
	QSYS	QSYSPRT	PRTF	CL source listing printer file. ¹
CRTBNDRPG	QGPL	QRPGLSRC	PF	ILE RPG/400 source default input file.
	QSYS	QSYSPRT	PRTF	ILE RPG/400 source listing printer file.
CRTCBLMOD	QGPL	QCBLLSRC	PF	ILE COBOL/400 source default input file.

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QSYSPRT	PF	ILE COBOL/400 source listing printer file.
CRTCBLPGM	QGPL	QLBLSRC	PF	COBOL/400 source default input file.
	QSYS	QSYSPRT	PRTF	COBOL/400 source listing printer file. ¹
CRTCLD	QGPL	QCLDSRC	PF	C locale description default source input file.
	QSYS	QSYSPRT	PRTF	C locale description source listing printer file. ¹
CRTCLMOD	QGPL	QCLSRC	PF	CL source default input file.
	QSYS	QSYSPRT	PRTF	CL source listing printer file. ¹
CRTCLPGM	QGPL	QCLSRC	PF	CL source default input file.
	QSYS	QSYSPRT	PRTF	CL source listing printer file. ¹
CRTCMD	QGPL	QCMSRC	PF	Command definition source default input file.
	QSYS	QSYSPRT	PRTF	Command definition source listing printer file.
CRTCMOD	QGPL	QCSRC	PF	C/400 source default input file.
	QSYS	QSYSPRT	PRTF	C/400 source listing printer file. ¹
CRTDSPF	QGPL	QDDSSRC	PF	DDS source default input file.
	QSYS	QPDDSSRC	PRTF	DDS source listing printer file. ¹
CRTICFF	QGPL	QDDSSRC	PF	DDS source default input file.
	QSYS	QPDDSSRC	PRTF	DDS source listing printer file. ¹
CRTLASREP	QGPL	QAOIASCI	PF	Default source physical file.
	QGPL	QAOIASCM	PF	Default source input file.
	QGPL	QAOIASCT	PF	Default metatable file.
	QOSI	QAOIXLAT	PF	Translation file.
CRTLFL	QGPL	QDDSSRC	PF	DDS source default input file.
	QSYS	QPDDSSRC	PRTF	DDS source listing printer file. ¹
CRTMNU	QGPL	QMNUSRC	PF	Default menu source input file.
	QSYS	QSYSPRT	PRTF	Menu source listing printer file. ¹
CRTMSGFMNU	QGPL	QDDSSRC	PF	DDS source created for menu by \$BMENU.
	QGPL	QS36DDSSRC	PF	DDS source created for menu by \$BMENU.
	QSSP	QPUTMENU	PRTF	\$BMENU source listing printer file.
	QSYS	QSYSPRT	PRTF	Pascal source listing printer file. ¹
CRTPF	QGPL	QDDSSRC	PF	DDS source default input file.
	QSYS	QPDDSSRC	PRTF	DDS source listing printer file. ¹
CRTPGM	QSYS	QSYSPRT	PRTF	Program source listing printer file. ¹
CRTPNLGRP	QGPL	QPNLSRC	PF	Default panel group source input file.
	QSYS	QSYSPRT	PRTF	Panel group source listing printer file. ¹
CRTPRTF	QGPL	QDDSSRC	PF	DDS source default input file.
	QSYS	QPDDSSRC	PRTF	DDS source listing printer file. ¹
CRTQSTDB	QSYS	QAQA00xxxx	LF	Q & A database model files. ²
	QSYS	QAQA00xxxx	PF	Q & A database model files. ²
CRTQSTLOD	QSYS	QPQAPRT	PRTF	Q & A printer file.
CRTRJESCF	QRJE	QRJESRC	PF	DDS source file used to create RJE BSC file.
CRTRJECFG	QRJE	QRJESRC	PF	DDS source file used to create RJE BSC or RJE communications file.
CRTRJECMNF	QRJE	QRJESRC	PF	DDS source file used to create RJE communications file.

Command Name	File Library	File Name	File Type	File Usage
CRTRPGMOD	QGPL	QRPGLESRC	PF	ILE RPG/400 source default input file.
	QSYS	QSYSPRT	PRTF	ILE RPG/400 source listing printer file. ¹
CRTRPGPGM	QGPL	QRPGSRC	PF	RPG source default input file.
	QSYS	QSYSPRT	PRTF	RPG source listing printer file. ¹
CRTRPTPGM	QGPL	QRPGSRC	PF	RPG source default input file.
	QSYS	QSYSPRT	PRTF	RPG source listing printer file. ¹
CRTSQLC	QGPL	QCSRC	PF	SQL C source file.
	QSYS	QSYSPRT	PRTF	SQL C printer file. ¹
CRTSQLCI	QGPL	QCSRC	PF	SQL C source file.
	QSYS	QSYSPRT	PRTF	SQL C printer file. ¹
CRTSQLCBL	QGPL	QLBLSRC	PF	SQL COBOL source file.
	QSYS	QSYSPRT	PRTF	SQL COBOL printer file. ¹
CRTSQLCBLI	QGPL	QCBLESRC	PF	SQL COBOL source file.
	QSYS	QSYSPRT	PRTF	SQL COBOL printer file. ¹
CRTSQLFTN	QGPL	QFTNSRC	PF	SQL FORTRAN source file.
	QSYS	QSYSPRT	PRTF	SQL FORTRAN printer file. ¹
CRTSQLPLI	QGPL	QPLISRC	PF	SQL PLI source file.
	QSYS	QSYSPRT	PRTF	SQL PLI printer file. ¹
CRTSQLRPG	QGPL	QRPGSRC	PF	SQL RPG source file.
	QSYS	QSYSPRT	PRTF	SQL RPG printer file. ¹
CRTSQLRPGI	QGPL	QRPGLESRC	PF	SQL RPG source file.
	QSYS	QSYSPRT	PRTF	SQL RPG printer file. ¹
CRTSRVPGM	QSYS	QSYSPRT	PRTF	Service program source listing printer file. ¹
CRTS36CBL	#LIBRARY	QS36SRC	PF	S/36-compatible COBOL source default input file.
	QSYS	QSYSPRT	PRTF	S/36-compatible COBOL source listing printer file. ¹
CRTS36DSPF	QGPL	QDDSSRC	PF	DDS source created for display file by \$SFGR.
	QGPL	QS36DDSSRC	PF	DDS source created for display file by \$SFGR.
CRTS36MNU	QSSP	QPUTSFGR	PRTF	\$SFGR source listing printer file.
	QGPL	QDDSSRC	PF	DDS source created for menu by \$BMENU.
CRTS36RPG	QGPL	QS36DDSSRC	PF	DDS source created for menu by \$BMENU.
	QSSP	QPUTMENU	PRTF	\$BMENU source listing printer file.
CRTS36RPG	#LIBRARY	QS36SRC	PF	System/36 RPG II source default input file.
	QSYS	QSYSPRT	PRTF	System/36 RPG II source listing printer file. ¹
CRTS36RPGR	#LIBRARY	QS36SRC	PF	System/36 RPG II source default input file.
CRTS36RPT	#LIBRARY	QS36SRC	PF	System/36 RPG II Auto Report source default input file.
	QSYS	QSYSPRT	PRTF	System/36 RPG II Auto Report source listing printer file. ¹
CRTTAPCGY	QUSRSYS	QATACGY	PF	Library device database file.
	QUSRSYS	QLTACGY	LF	Library device logical database file.
CRTTBL	QGPL	QTBLSRC	PF	Table source default input file.

Command Name	File Library	File Name	File Type	File Usage
CVTPFRDTA	QPFR	QACPxxxx	PF	QACPxxxx performance data collection files, where xxxx = CNFG, GPHF, PROF, and RESP.
	QPFR	QAITMON	PF	Performance data collection file: job and Licensed Internal Code task data.
	QSYS	QAPMyyyy	PF	QAPMyyyy performance data collection files, where yyyy = CIOP, CONF, DIOP, DISK, JOBS, LIOP, POOL, RESP, RWS, and SYS. ²
	QSYS	QAPMzzzz	PF	QAPMzzzz performance data collection files, where zzzz = ASYN, BSC, BUS, DMPT, ECL, ETH, HDLC, MIOF, and X25. ²
	QPFR	QAPTAPGP	PF	Performance data collection file for functional area data.
CVTRPGSRC	user-lib	QRPGSRC	PF	RPG/400 source file (from file).
	user-lib	QRPGLESRC	PF	ILE RPG/400 source file (to file).
	QSYS	QSYSPRT	PRTF	ILE RPG/400 listing printer file.
DLTALR	user-lib	QRNCVTLG	PF	ILE RPG/400 conversion log file.
	QUSRSYS	QAALERT	PF	Alert database file.
DLTBESTMDL	user-lib	QACYMDLS	PF	BEST/1 file of capacity planning models.
DLTPFRDTA	QPFR	QAPGSUMD	PF	Performance data collection file for graphics data.
	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = ASYN, BSC, BUS, CIOP, CONF, DIOP, DISK, DMPT, ECL, ETH, HDLC, IOBS, JOBS, LIOP, MIOF, POOL, RESP, RWS, SBSO, SYS, TSK, and X25. ²
DLTPFRDTA	QPFR	QAPTLCKD	PF	Performance data collection file: lock and seizure conflict data.
	QPFR	QTRxxxx	PF	QTRxxxx performance data collection files, where xxxxx = IDX, JOBT, JSUM, SLWT, and TSUM.

DLTPRB Command: All 8 of the QASXxxxx files for the DLTPRB command are the same subset of files that are shown in the QUSRSYS library for the DSPPRB command. For the descriptions of these files, see the DSPPRB command.

DLTPRB	QUSRSYS	QASXxxxx	PF	See the DSPPRB command.
DLTQST	QSYS	QPQAPRT	PRTF	Q & A printer file.
DLTQSTDB	QSYS	QPQAPRT	PRTF	Q & A printer file.
DLTTAPCGY	QUSRSYS	QATACGY	PF	Library device database file.
	QUSRSYS	QLTACGY	LF	Library device logical database file.
DMPCLPGM	QSYS	QPPGMDMP	PRTF	CL program dump printer file.
DMPJOB	QSYS	QPSRVDMP	PRTF	Service dump printer file.
DMPOBJ	QSYS	QPSRVDMP	PRTF	Service dump printer file.
DMPYSOBB	QSYS	QPSRVDMP	PRTF	Service dump printer file.
DMPTAP	QSYS	QPTAPDMP	PRTF	Tape dump printer file.
DMPTRC	QSYS	QAPMDMPT	PF	Performance trace file.
DSPACC	QSYS	QSYSPRT	PRTF	SDA source printer file.
	QSYS	QSYSPRT	PRTF	Display access codes printer file. ¹

Command Name	File Library	File Name	File Type	File Usage
DSPACCAUT	QSYS	QSYSPRT	PRTF	Display access code authority printer file. ¹
DSPACCGRP	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QAPTPAGD	PF	Performance data collection file: data about files and programs in a process access group (PAG).
	QPFR	QPPTPAG	PRTF	Display process access group printer file. ¹
	QPFR	QPPTPAGD	PRTF	Performance printer file containing files and programs in a process access group (PAG).
DSPACTPJ	QSYS	QSYSPRT	PRTF	Display active prestart jobs printer file. ¹
DSPAPPNINF	QUSRSYS	QALSxxx	PF	Set of 4 QALSxxx model database files that contain the record formats used for storing APPN information, where xxx = DIR, END, INM, and TDB. ²
	QSYS	QSYSPRT	PRTF	Display APPN information printer file. ¹
DSPAUTHLR	QSYS	QADSHLR	PF	Model database file that contains the record format for the authority holder object entries. ²
DSPAUTL	QSYS	QPSYDSHL	PRTF	Display authority holders printer file. ¹
	QSYS	QAOBJAUT	PF	Model database file that contains the record format for the authorization list entries. ²
	QSYS	QPOBJAUT	PRTF	Authorization list entries printer file. ¹
DSPAUTLDLO	QSYS	QSYSPRT	PRTF	Authorization list printer file. ¹
DSPAUTLOBJ	QSYS	QADALO	PF	Model database file that contains the record format for the authorization list object entries. ²
	QSYS	QPSYDALO	PRTF	Display authorization list objects printer file. ¹
DSPAUTUSR	QSYS	QPAUTUSR	PRTF	Authorized users printer file. ¹
DSPBCKSTS	QSYS	QSYSPRT	PRTF	Display backup status printer file. ¹
DSPBCKUP	QSYS	QSYSPRT	PRTF	Display backup options printer file. ¹
DSPBCKUPL	QSYS	QSYSPRT	PRTF	Display backup list printer file. ¹
DSPBKP	QSYS	QPDBGDSP	PRTF	Breakpoint (debug mode) printer file. ¹
DSPBNDDIR	QSYS	QABNDBND	PF	Model database file that contains the record format for the binding directory entries. ²
	QSYS	QSYSPRT	PRTF	Displays the contents of the binding directory printer file. ¹
DSPCFGL	QSYS	QPDCCFGL	PRTF	Configuration list printer file. ¹
DSPCHT	QSYS	QPGDDM	PRTF	BGU-defined chart output printer file. ¹
DSPCLS	QSYS	QPDSPCLS	PRTF	Class printer file. ¹
DSPCMD	QSYS	QPCMD	PRTF	Command values printer file. ¹
DSPCNNL	QSYS	QPDCCNNL	PRTF	Connection list printer file. ¹
DSPCNNSTS	QSYS	QSYSPRT	PRTF	Connection status printer file. ¹
DSPCOSD	QSYS	QPDCCOS	PRTF	Class-of-service description printer file. ¹
DSPCSI	QSYS	QSYSPRT	PRTF	Communications side information printer file. ¹
DSPCTLD	QSYS	QPDCCTL	PRTF	Controller description printer file. ¹

Command Name	File Library	File Name	File Type	File Usage
DSPDBG	QSYS	QPDBGDSP	PRTF	Debug display (debug mode) printer file. ¹
DSPDBR	QSYS	QADSPDBR	PF	Model database file that defines the record format of the file created to store information about database file relationships. ²
	QSYS	QPDSPDBR	PRTF	Printer file that contains information about database file relationships. ¹
DSPDDMF	QSYS	QPDSPDDM	PRTF	Distributed data management (DDM) file listing printer file. ¹
DSPDEVD	QSYS	QPDCDEV	PRTF	Device description printer file. ¹
	QSYS	QAOSDIRO	PF	Display directory output file: OUTFILFMT(*TYPE1)
	QSYS	QAOSDIRB	PF	Display directory output file: OUTFILFMT(*TYPE2) DETAIL(*BASIC)
	QSYS	QAOSDIRF	PF	Display directory output file: OUTFILFMT(*TYPE2) DETAIL(*FULL)
	QSYS	QAOSDIRX	PF	Display directory output file: OUTFILFMT(*TYPE3) DETAIL(*FULL)
	QSYS	QPDSPDDL	PRTF	Printer file for <i>full</i> details of displayed directory entries. ¹
	QSYS	QPDSPDSM	PRTF	Printer file for <i>basic</i> details of displayed directory entries. ¹
DSPDKT	QSYS	QPDSPDKT	PRTF	Printer file for diskettes in basic data exchange format. ¹
	QSYS	QPSRODSP	PRTF	Printer file for diskettes in save/restore format. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file for input.
DSPDLOAUT	QSYS	QSYSPRT	PRTF	Display document library object authority printer file. ¹
DSPDLONAM	QSYS	QSYSPRT	PRTF	Display document library object printer file. ¹
DSPDSTL	QSYS	QAOSDSTO	PF	Distribution list output file.
	QSYS	QPDSPLDL	PRTF	Distribution list <i>details</i> printer file. ¹
	QSYS	QPDSPLSM	PRTF	Distribution list <i>summary</i> printer file. ¹
DSPDSTCLGE	QSVMS	QACQFVOF	PF	Output file model for MSS/400 Display Distribution Catalog Entries command. ²
DSPDSTLOG	QSYS	QPDSTDLG	PRTF	Display distribution log printer file. ¹
DSPDSTSRV	QSYS	QPDSTSRV	PRTF	Distribution services printer file. ¹
	QUSRSYS	QASNADSA	PF	SNADS secondary node ID table.
	QUSRSYS	QASNADSQ	PF	SNADS destination queues table.
	QUSRSYS	QASNADSR	PF	Routing table database file.
DSPDTA	QIDU	QDTAPRT	PRTF	DFU audit control printer file.
	QSYS	QPDZDTALOG	PRTF	DFU run-time audit log.
	QSYS	QPDZDTAPRT	PRTF	DFU run-time printer data file.
DSPDTAARA	QSYS	QPDSPTA	PRTF	Data area printer file. ¹
DSPDTADCT	QSYS	QPDSPPFD	PRTF	Data dictionary printer file. ¹
DSPEDTD	QSYS	QPDCEDSP	PRTF	Edit description printer file. ¹


Command Name	File Library	File Name	File Type	File Usage
DSPFD Command:				For the DSPFD command, all of the following entries having a file type of PF are physical files (model files that are not actual output files) that define the record formats of created files used to store a specific type of information about a type (or group) of files. (See the DSPFD command for descriptions of the TYPE and FILEATR parameters. These parameters identify all the values that cause these files to be used.) Therefore, the common part of each model file's description <i>begins</i> here and the unique part of each description <i>continues</i> under the File Usage column:
				Model database file that defines the record format of the file created to store...
DSPFD	QSYS	QAFDACC	PF	...access path file information. ²
	QSYS	QAFDBASI	PF	...basic file information common to all files. ²
	QSYS	QAFDBSC	PF	...BSC file and mixed file device attribute information. ²
	QSYS	QAFDCMN	PF	...communications file and mixed file device attribute information. ²
	QSYS	QAFDCSEQ	PF	... collating sequence information. ²
	QSYS	QAFDCST	PF	... constraint relationship information. ²
	QSYS	QAFDDDM	PF	...distributed data management (DDM) file attribute information. ²
	QSYS	QAFDDKT	PF	...diskette file attribute information. ²
	QSYS	QAFDDSP	PF	...display file and mixed file display device attribute information. ²
	QSYS	QAFDICF	PF	...ICF file attribute information. ²
	QSYS	QAFDJOIN	PF	...join logical file information. ²
	QSYS	QAFDLGL	PF	...logical file attribute information. ²
	QSYS	QAFDMBR	PF	...database member information. ²
	QSYS	QAFDMBRL	PF	...database member list information. ²
	QSYS	QAFDNGP	PF	... node group information. ²
	QSYS	QAFDPHY	PF	...physical file attribute information. ²
	QSYS	QAFDPRT	PF	...printer file attribute information. ²
	QSYS	QAFDRFMT	PF	...record format information. ²
	QSYS	QAFDSAV	PF	...save file information. ²
	QSYS	QAFDSELO	PF	...select/omit information. ²
	QSYS	QAFDSPOL	PF	...device file spooled information. ²
	QSYS	QAFDTAP	PF	...tape file attribute information. ²
	QSYS	QAFDTRG	PF	... trigger information. ²
	QSYS	QPDSFPFD	PRTF	File description printer file. ¹
DSPFFD	QSYS	QADSPFFD	PF	Model database file that defines the record format of the file created to store file field descriptions. ²
	QSYS	QPDSPPFD	PRTF	File field description printer file. ¹
DSPFLR	QSYS	QADSPDOC	PF	Document list output database file.
	QSYS	QADSPFLR	PF	Folder list output database file.
	QSYS	QPDSPPFLR	PRTF	Display folder printer file. ¹
DSPFNTRSCA	QSYS	QPDSPPFNT	PRTF	Font resource attributes printer file. ¹
DSPGDF	QSYS	QPGDDM	PRTF	BGU-defined graphics data printer file. ¹
DSPGNDDIR	QSYS	QABNDBND	PF	System-supplied output file. ¹

Command Name	File Library	File Name	File Type	File Usage
DSPHDWRSC Command:	For the DSPHDWRSC command, all of the following entries having a file type of PF are physical files (model files that are not actual output files) that define the record formats of created files used to store a specific type of hardware resource information. Therefore, the common part of each model file's description <i>begins</i> here and the unique part of each description <i>continues</i> under the File Usage column:			

Model database file that defines the record format of the file created to store information about...

DSPHDWRSC	QSYS	QARZDCMN	PF	...communications resources. ²
	QSYS	QARZDLWS	PF	...local work station resources. ²
	QSYS	QARZDPRC	PF	...processor resources. ²
	QSYS	QARZDTRA	PF	...token-ring local area network (TRLAN) adapter resources. ²
	QSYS	QARZDSTG	PF	...storage device resources. ²
	QSYS	QSYSPRT	PRTF	Hardware resources printer file. ¹
DSPHFS	QSYS	QSYSPRT	PRTF	Display hierarchical file systems printer file. ¹
DSPHSTGPH	QPFR	QPPGGPH	PRTF	Display historical graph printer file. ¹
DSPIDXSTS	QUSRSYS	QABBADMTB	PF	OfficeVision text search services administration table.
DSPIGCDCT	QSYS	QPDPSPDCT	PRTF	DBCS printer file. ¹
DSPJOB	QSYS	QPDPSPJOB	PRTF	Display job printer file. ¹
DSPJOBBD	QSYS	QPRTJOBBD	PRTF	Job description printer file. ¹
DSPJOBLOG	QSYS	QPJOBLOG	PRTF	Job log printer file. ¹
	QSYS	QPJOBLOGO	PRTF	Job log printer file for jobs prior to Version 2 Release 3 of the AS/400. ¹
	QSYS	QAMHJLPR	PF	Primary job log model file. ²
	QSYS	QAMHJLSC	PF	Secondary job log model file. ²

DSPJRN Command: All of the following files for the DSPJRN command having a file type of PF are physical files (model files, not actual output files) that define the record formats of files created to store a group of journal entries retrieved and converted from journal receivers. The retrieved group of entries can be a specific type of information or all types of information that was journaled. Each created file stores the retrieved journal entries after they are converted into one of four basic formats (*TYPE1, *TYPE2, *TYPE3 or *TYPE4) or into the format defined for the specific type of data being retrieved.

- *TYPE1: the basic file format described in the Backup and Recovery  book.
- *TYPE2: all of *TYPE1 plus the user profile field.
- *TYPE3: all of *TYPE2 plus the null value indicators.
- *TYPE4: all of *TYPE3 plus the JID, referential integrity and trigger information.
- Type-dependent format - the format associated with the specific type (as described below for the fourth and following files) of information being retrieved. For example, the model file QASYAFJE has a unique format for storing all retrieved journal entries related to authority failures (AF) on the system.

For the DSPJRN PF files listed below, the common part of all the model files' descriptions begins here and the unique part of each file's description continues under the **File Usage** column:

Model database file that defines the record format of the file created to contain retrieved and converted journal entries related to...

DSPJRN	QSYS	QADSPJRN	PF	...a specific type (or all types) of information; stored in the *TYPE1 format. ²
--------	------	----------	----	---

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QADSPJR2	PF	...a specific type (or all types) of information; stored in the *TYPE2 format. ²
	QSYS	QADSPJR3	PF	...a specific type (or all types) of information; stored in the *TYPE3 format. ²
	QSYS	QADSPJR4	PF	...a specific type (or all types) of information; stored in the *TYPE4 format. ²
	QSYS	QADXERLG	PF	...DSNX logged errors. ²
	QSYS	QADXJRNL	PF	...DSNX logged data. ²
	QSYS	QAJBACG	PF	...job accounting. ²
	QSYS	QALZALK	PF	...invalid license keys. ²
	QSYS	QALZALL	PF	...usage limit increases. ²
	QSYS	QALZALU	PF	...licensed users that exceed usage limits. ²
	QSYS	QAPTACG	PF	...print job accounting. ²
	QSYS	QATOSLOG	PF	...SNMP log entries. ²
	QSYS	QASYADJE	PF	...changes to auditing attributes. ²
	QSYS	QASYAFJE	PF	...authority failures. ²
	QSYS	QASYAPJE	PF	...use of adopted authority. ²
	QSYS	QASYCAJE	PF	...changes to object authority (authorization list or object). ²
	QSYS	QASYCDJE	PF	...command strings. ²
	QSYS	QASYCOJE	PF	...objects created on the system. ²
	QSYS	QASYCPJE	PF	...create, change, and restore user profile operations. ²
	QSYS	QASYCQJE	PF	...changes to *CRQD object. ²
	QSYS	QASYDOJE	PF	...objects deleted from the system. ²
	QSYS	QASYDSJE	PF	...resetting the DST security officer password. ²
	QSYS	QASYGSJE	PF	...gives of descriptors. ²
	QSYS	QASYIPJE	PF	...interprocess communications. ²
	QSYS	QASYJDJE	PF	...changes to the USER parameter of job descriptions. ²
	QSYS	QASYJSJE	PF	...job changes. ²
	QSYS	QASYLDJE	PF	...link/unlink/lookup directory. ²
	QSYS	QASYMLJE	PF	...mail actions. ²
	QSYS	QASYNAJE	PF	...changes to network attributes. ²
	QSYS	QASYNDJE	PF	...directory search violations. ²
	QSYS	QASYNEJE	PF	...end point violations. ²
	QSYS	QASYOMJE	PF	...object move and rename operations. ²
	QSYS	QASYORJE	PF	...object restore operations. ²
	QSYS	QASYOWJE	PF	...changes to object ownership. ²
	QSYS	QASYO1JE	PF	...single optical object accesses. ²
	QSYS	QASYO2JE	PF	...dual optical object accesses. ²
	QSYS	QASYO3JE	PF	...optical volume accesses. ²
	QSYS	QASYPAJE	PF	...changes to programs (CHGPGM) that will now adopt the owner's authority. ²
	QSYS	QASYPGJE	PF	...changes to object primary group. ²
	QSYS	QASYPOJE	PF	...printer output actions. ²
	QSYS	QASYPSJE	PF	...profile swapping. ²

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QASYPWJE	PF	...attempted usage of invalid passwords or user profile names. ²
	QSYS	QASYRAJE	PF	...restore of objects when authority changes. ²
	QSYS	QASYRJJE	PF	...restore of job descriptions that contain user profile names. ²
	QSYS	QASYROJE	PF	...restore of objects when ownership was changed to QDFTOWN. ²
	QSYS	QASYRPJE	PF	...restore of programs that adopt their owner's authority. ²
	QSYS	QASYRQJE	PF	...restores of *CRQD object. ²
	QSYS	QASYRUJE	PF	...operations restoring authority for user profiles, using the RSTAUT command. ²
	QSYS	QASYRZJE	PF	...changes on primary group restores. ²
	QSYS	QASYSDJE	PF	...changes to the system distribution directory. ²
	QSYS	QASYSEJE	PF	...changes to subsystem routings. ²
	QSYS	QASYSFJE	PF	...actions with spooled files. ²
	QSYS	QASYSMJE	PF	...system management changes. ²
	QSYS	QASYSOJE	PF	...server security changes. ²
	QSYS	QASYSTJE	PF	...use of system service tools. ²
	QSYS	QASYSVJE	PF	...changes to system values. ²
	QSYS	QASYVAJE	PF	...changes to access control list. ²
	QSYS	QASYVCJE	PF	...connection starts and ends. ²
	QSYS	QASYVFJE	PF	...closes of server files. ²
	QSYS	QASYVLJE	PF	...exceeding account limit. ²
	QSYS	QASYVNJE	PF	...network logons and logoffs. ²
	QSYS	QASYVPJE	PF	...network password errors. ²
	QSYS	QASYVRJE	PF	...accesses to network resources. ²
	QSYS	QASYVSJE	PF	...server session starts and ends. ²
	QSYS	QASYVUJE	PF	...changes to network profiles. ²
	QSYS	QASYVVJE	PF	...changes to service status. ²
	QSYS	QASYVCJE	PF	...changes to document library objects. ²
	QSYS	QASYRJE	PF	...read operations of document library objects. ²
	QSYS	QASYZCJE	PF	...changes to objects. ²
	QSYS	QASYZMJE	PF	...object method access. ²
	QSYS	QASYZRJE	PF	...read operations of objects. ²
	QSYS	QAWCTPJE	PF	...performance tuning. ²
	QSYS	QAZDCFLG	PF	...configuration changes to SNADS distribution queues table. ²
	QSYS	QAZDERLG	PF	...SNADS logged errors. ²
	QSYS	QAZDJRNL	PF	...SNADS logged data. ²
	QSYS	QAZDRTL	PF	...changes to SNADS routing and secondary system name tables. ²
	QSYS	QAZDSYLG	PF	...SNADS miscellaneous logged system-level occurrences. ²
	QSYS	QAZMFCF	PF	...Mail server framework (MSF) configuration changes logging. ²
	QSYS	QAZMFER	PF	...Mail server framework (MSF) error logging. ²

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QAZMFLG	PF	...Mail server framework (MSF) data logging. ²
	QSYS	QAZMFSY	PF	...Mail server framework (MSF) system information logging. ²
	QSYS	QPDSPJRN	PRTF	Display journal printer file. ¹
	QX400	QAX4ELOG	PF	...X.400 logged errors. ²
	QX400	QAX4MLOG	PF	...changes to X.400 message transfer agent configurations. ²
	QX400	QAX4NLOG	PF	...X.400 notifications of delivery or nondelivery. ²
	QX400	QAX4RLOG	PF	...changes to X.400 route configurations. ²
	QX400	QAX4ULOG	PF	...X.400 user or probe messages. ²
DSPJRNRCVA	QSYS	QPDSPRCV	PRTF	Journal receiver attributes printer file. ¹
	QSYS	QSYSPRT	PRTF	Local hardware resources printer file. ¹
DSPLIB	QSYS	QPDSPLIB	PRTF	Library printer file. ¹
DSPLIBD	QSYS	QPRTLIBD	PRTF	Library description printer file. ¹
DSPLIBL	QSYS	QPRTLIBL	PRTF	Library list printer file. ¹
DSPLIND	QSYS	QPDCLINE	PRTF	Line description printer file. ¹
DSPLOG	QSYS	QPDSPLG	PRTF	Log display printer file. ¹
DSPMNUA	QSYS	QPDSPMNU	PRTF	Menu attributes printer file. ¹

DSPMOD Command:

For the DSPMOD command, all of the following entries having a file type of files used to store a specific type of information about a type (or group) of files. (See the DSPMOD command for a description of the DETAIL parameter. This parameter identifies all the values that cause these files to be used.) Therefore, the common part of each model file's description *begins* here and the unique part of each description *continues* under the **File Usage** column:

Model database file that defines the record format of the file created to store...

DSPMOD	QSVMS	QACQSRC	PRTF	...contains sources of example security exit programs. ¹
	QSYS	QABNDMBA	PF	...basic information and compatibility sections. ¹
	QSYS	QABNDMSI	PF	...decompressed size and size limits. ¹
	QSYS	QABNDMEX	PF	...symbols defined in this module and exported to others. ¹
	QSYS	QABNDMIM	PF	...defined symbols that are external to this module. ¹
	QSYS	QABNDMPR	PF	...a list of procedure names and types. ¹
	QSYS	QABNDMRE	PF	...a list of system objects referred to by the module at the time the module is bound into a program or service program. ¹
	QSYS	QABNDMCO	PF	...module copyright information.
	QSYS	QSYSPRT	PRTF	...module printer file. ¹
DSPMODD	QSYS	QPDCMOD	PRTF	Mode description printer file. ¹
DSPMODSTS	QSYS	QPDCMOD	PRTF	Mode status printer file. ¹
DSPMSG	QSYS	QPDSMSG	PRTF	Message display printer file. ¹
DSPMSGD	QSYS	QPMSGD	PRTF	Message description printer file. ¹

Command Name	File Library	File Name	File Type	File Usage
DSPNETA	QSYS	QANFDNTF	PF	Model database file used to define the record format for network file entries. ²
	QSYS	QPDISPNET	PRTF	Display network attributes printer file. ¹
	QSYS	QPNFNJE	PRTF	Display network job entries printer file. ¹
DSPNWID	QSYS	QPDCNWID	PRTF	Network interface description printer file. ¹
DSPOBJAUT	QSYS	QAOBJAUT	PF	Model database file that defines the record format for the object authority entries. ²
	QSYS	QPOBJAUT	PRTF	Object authority printer file. ¹
DSPOBJD	QSYS	QADSPOBJ	PF	Model database file that defines the record format for the object description entries. ²
	QSYS	QPRTOBJD	PRTF	Object description printer file. ¹
DSPOVR	QSYS	QPDSPOVR	PRTF	Display overrides printer file. ¹
DSPPDGPRF	QGPL	QPCJPDGPRF	PRTF	Printer file for print descriptor group profile. ¹
	QPFRDATA	QAPGGPHF	PF	Performance database file: graph format data.
	QPFRDATA	QAPGPKGf	PF	Performance database file: graph package data.
	QPFR	QPPGGPH	PRTF	Performance graphs printer file. ¹
DSPPGM	QSYS	QPDPGM	PRTF	Display program printer file. ¹
DSPPGMADP	QSYS	QADPGMAD	PF	Model database file that defines the record format of the file created to store the names of programs that adopt the specified profile. ²
	QSYS	QPPGMADP	PRTF	Printer file that lists the programs that adopt the specified profile. ¹
DSPPGMREF	QSYS	QADSPPGM	PF	Model database file that defines the record format of the file created to store program references. ²
	QSYS	QPDSPPGM	PRTF	Printer file that contains program references. ¹
DSPPGMVAR	QSYS	QPDBGDSP	PRTF	Program variable (debug mode) printer file. ¹

DSPPRB Command: All 8 of the QASXxxxx files shown below in the QUSRSYS library are also used by the DLTPRB and WRKPRB commands. The other files below (in QSYS) are *not* used by those commands.

DSPPRB	QSYS	QASXxxxx	PF	Set of 5 QASXxxxx model database files that contain the layouts for problem output files, where xxxx = CAOF, FXOF, PBOF, SDOF, and TXOF. ²
	QSYS	QSXPRTD	PRTF	Problem log <i>detail</i> printer file. ¹
	QSYS	QSXPRTL	PRTF	Problem log <i>summary</i> printer file. ¹

³ The following 8 files are also used by the DLTPRB and WRKPRB commands.

QUSRSYS	QASXCALL ³	PF	Problem log call override file.
QUSRSYS	QASXDTA ³	PF	Problem log data identifier file.
QUSRSYS	QASXEVT ³	PF	Problem log event file.
QUSRSYS	QASXFRU ³	PF	Problem log possible cause file.
QUSRSYS	QASXNOTE ³	PF	Problem log user notes file.

Command Name	File Library	File Name	File Type	File Usage
DSPPTF	QUSRSYS	QASXPROB ³	PF	Problem log problem file.
	QUSRSYS	QASXPTF ³	PF	Problem log PTF file.
	QUSRSYS	QASXSYMP ³	PF	Problem log symptom string file.
	QSYS	QADSPPTF	PF	Model database file that defines the record format of the file created to store program temporary fix (PTF) information. ²
	QSYS	QSYSPRT	PRTF	Display programming temporary fix (PTF) printer file. ¹
DSPPWRSCD	QSYS	QSYSPRT	PRTF	Display power schedule printer file. ¹
DSPRCDLCK	QSYS	QPDSPLK	PRTF	Record locks display printer file. ¹
DSPRDBDIRE	QSYS	QSYSPRT	PRTF	Distributed relational database directory printer file. ¹
	QSYS	QADSPDE	PF	Model database file that defines the record format for the RDB directory entries.
	QRJE	QPRTCFG	PRTF	RJE configuration printer file. ¹
DSPSAVF	QSYS	QPSRODSP	PRTF	Printer file for save file save/restore information. ¹
DSPSBSD	QSYS	QPRTSBSD	PRTF	Subsystem description printer file. ¹
DSPSFWRSC	QSYS	QARZLCOF	PF	Model database file of the file created to store information about IBM licensed programs and SystemView packaged applications. ²
	QSYS	QSYSPRT	PRTF	Software resources printer file. ¹
DSPSOCSTS	QSYS	QSYSPRT	PRTF	Sphere of control status printer file. ¹
	QUSRSYS	QAALSOC	PF	Sphere of control database file.
DSPSRVPGM	QSYS	QSYSPRT	PRTF	Service program printer file. ¹
DSPSYSSTS	QSYS	QPDSPSTS	PRTF	Display system status printer file. ¹
DSPSYSVAL	QSYS	QPDSPSVL	PRTF	System value printer file. ¹
DSPTAP	QSYS	QPTAPDSP	PRTF	Printer file for tape output. ¹
	QSYS	QPSRODSP	PRTF	Printer file for tapes in save/restore format. ¹
	QSYS	QSYSTAP	TAPF	Tape device file for input.
DSPTAPCGY	QSYS	QTAPCGY	PRTF	Printer file for tape categories 1.
	QSYS	QATACOF	PF	Model output file for tape categories 2.
	QUSRSYS	QATACGY	PF	Library device database file.
	QUSRSYS	QLTACGY	LF	Library device logical database file.
DSPTAPCTG	QSYS	QPTACTG	PRTF	Printer file for tape cartridge identifiers 1.
	QSYS	QATAVOF	PF	Model output file for tape cartridge identifiers 2.
	QUSRSYS	QATAMID	PF	Library device database file.
	QUSRSYS	QATAMID	LF	Library device logical database file.
DSPTAPSTS	QSYS	QPTAPSTS	PRTF	Printer file for tape library
	QSYS	QATAIOF	PF	Model output file for tape
	QSYS	QSYSTAP	TAPF	Tape device file or input
	QSYS	QPTYSWTD	PRTF	Printer file for printing a telephony switch entry. ¹
DSPTRAPRF	QSYS	QSYSPRT	PRTF	Printer file containing a token-ring network adapter profile. ¹
DSPTRC	QSYS	QPDBGDSP	PRTF	Trace (debug mode) printer file. ¹
DSPTRCDTA	QSYS	QPDBGDSP	PRTF	Trace data (debug mode) printer file. ¹

Command Name	File Library	File Name	File Type	File Usage
DSPTRNSTS	QSYS	QSYSPRT	PRTF	Token-ring network status printer file. ¹
DSPUSRPMN	QSYS	QSYSPRT	PRTF	Display document authority printer file. ¹
DSPUSRPRF	QSYS	QADSPUPA	PF	Model database file that defines the record format of user profiles for TYPE(*OBJAUT). ²
	QSYS	QADSPUPB	PF	Model database file that defines the record format of user profiles for TYPE(*BASIC). ²
	QSYS	QADSPUPO	PF	Model database file that defines the record format of user profiles for TYPE(*OBJOWN). ²
DSPWSUSR	QSYS	QPUSRPRF	PRTF	User profile printer file. ¹
	QSYS	QSYSPRT	PRTF	Work station user printer file. ¹
DUPTAP	QSYS	QSYSTAP	TAPF	Tape device file used for input and output.
EDTIGCDCT	QSYS	QPDSPDCT	PRTF	DBCS printer file.
EDTQST	QSYS	QPQAPRT	PRTF	Q & A printer file.
EJTEMLOUT	QSYS	QPEMPRTF	PRTF	Emulation printer file.
ENDDSKCOL	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QAPTDSKD	PF	Performance data collection file: disk activity data.
ENDIDXMON	QUSRSYS	QABBADMTB	PF	OfficeVision text search services administration table.
ENDJOBTRC	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QAPTTRCJ	PF	Performance data collection file: job trace data.
	QPFR	QPPTTRCD	PRTF	Performance printer file containing job trace analysis <i>detail</i> data.
	QPFR	QPPTTRC1	PRTF	Performance printer file containing job trace analysis <i>summary</i> data of physical disk activity.
ENDPRTEML	QPFR	QPPTTRC2	PRTF	Performance printer file containing job trace analysis I/O <i>summary</i> data.
	QSYS	QPEMPRTF	PRTF	Emulation printer file.
	QPFR	QAPTSAMH	PF	Performance data collection file: high-level sampled address monitor (SAM) data.
FMTDTA	QPFR	QAPTSAMV	PF	Performance data collection file: low-level sampled address monitor (SAM) data.
	QSYS	QSYSPRT	PRTF	Data format printer file.
FNDSTRPART	QGPL	QFMTSRC	PF	Sort source default input file.
	QPDA	QPUOPRTF	PRTF	PDM printer output file for user's find string requests.
FNDSTRPDM	QPDA	QPUOPRTF	PRTF	PDM printer output file for user's find string requests.
HLDDSTQ	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.
INZDKT	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
INZTAP	QSYS	QSYSTAP	TAPF	Tape device file used for output.
LODPTF	QSYS	QSYSDKT	DKTF	Diskette device file used for input.
LODQSTDB	QSYS	QSYSTAP	TAPF	Tape device file used for input.
	QQALIB	QAQAxxxx00	PF	Q & A supplied model database files. ²

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QAQA00xxxx	LF	Q & A database model files. ²
	QSYS	QAQA00xxxx	PF	Q & A database model files. ²
	QSYS	QPQAPRT	PRTF	Q & A printer file.
MRGFORMD	QPDA	QPAPFPRT	PRTF	Merge form description printer file.
MRGTCPTH	QUSRSYS	QATOCHOST	PF	TCP/IP hosts file.
PRTACTRPT	user-lib	QAITMON	PF	Performance data collection file: job and Licensed Internal Code task data.
	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPITACTR	PRTF	Performance printer file containing job and Licensed Internal Code task data.
PRTAFPDTA	QSYS	QSYSPRT	PRTF	Advanced Function Printing data printer file.
PRTCMDUSG	QSYS	QSYSPRT	PRTF	Command usage printer file.
PRTCMNTRC	QSYS	QASCCMNT	PF	Model database file that defines the record format of the file created to store communications trace records. ²
	QSYS	QPCSMPT	PRTF	Communications trace printer file (concurrent service monitor). ¹
PRTCPTRPT	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPPTCPT	PRTF	Performance printer file containing component-level activity data.
	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = ASYN, BSC, CIOP, CONF, DIOP, DISK, ECL, ETH, HDLC, JOBS, LIOP, MIO, POOL, RESP, RWS, and SYS. ²
PRTDEVADR	QSYS	QPDDEVA	PRTF	Print device address printer file.
PRTDOC	QSYS	QAPOUFL	PF	Document output database file. ¹
	QSYS	QSYSPRT	PRTF	Print document printer file. ¹
PRTDSKINF	QSYS	QAEZDISK	PF	Model outfile for disk space information.
	QUSRSYS	QAEZDISK	PF	Database input file of disk space information.
	QSYS	QPEZDISK	PRTF	Disk space report printer file.
	QSYS	QSYSPRT	PRTF	Disk space report printer file. This file must be specified if using OVRPRTF.
PRTDSKRPT	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QAPTDSKD	PF	Performance database input file of disk activity collection data.
	QPFR	QPPTDSK	PRTF	Printer file of disk unit I/O activity data.
PRTERLOG	QSYS	QAPRTELG	PF	Model database file that defines the record format of the file created to store error log records. ²
	QSYS	QAVOLSTA	PF	Model database file that defines the record format of the file created to store volume statistics. ²
	QSYS	QPCSMPT	PRTF	Error log (for concurrent service monitor) printer file. ¹
PRTINTDTA	QSYS	QPCSMPT	PRTF	Internal data (for concurrent service monitor) printer file.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
PRTJOBRT	QPFR	QAPTDDS	PF	Performance data DDS source file.

Command Name	File Library	File Name	File Type	File Usage
	QPFR	QPPTITVJ	PRTF	Performance printer file containing job interval collection data.
	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = CONF and JOBS. ²
PRTJOBTRC	QPFR	QAJOBTRC	PF	Performance data collection file: job trace data.
	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QAPTTRCJ	PF	Performance data collection file: job trace data.
	QPFR	QPPTTRCD	PRTF	Performance printer file containing job trace analysis <i>detail</i> data.
	QPFR	QPPTTRC1	PRTF	Performance printer file containing job trace analysis <i>summary</i> data of physical disk activity.
	QPFR	QPPTTRC2	PRTF	Performance printer file containing job trace analysis I/O <i>summary</i> data.
PRTLCKRPT	user-lib	QAPTLCKD	PF	Performance data collection <i>output</i> file containing lock and seizure conflict data.
	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPPTLCK	PRTF	Performance data collection <i>printer</i> file containing lock and seizure conflict data.
	QSYS	QAPMDMPT	PF	Performance data collection <i>input</i> file containing system lock and seizure conflict trace data. ²
PRTPOLRPT	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPPTITVP	PRTF	Performance printer file containing subsystem and pool activity interval data.
	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = CONF, JOBS, and POOL. ²
PRTRSCRPT	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPPTITVR	PRTF	Performance printer file containing disk and communications line activity interval data.
	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = ASYN, BSC, CIOP, CONF, DIOP, DISK, ECL, ETH, HDLC, LIOP, MIOP, RESP, RWS, and SYS. ²
	QPFR	QAPTSAMH	PF	Performance data collection file: high-level sampled address monitor (SAM) data.
	QPFR	QAPTSAMV	PF	Performance data collection file: low-level sampled address monitor (SAM) data.
	QPFR	QPPTSAM	PRTF	Printer file of SAM performance data.
PRTSYSRPT	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QPPTSYSR	PRTF	Performance printer file containing system workload and resource utilization data.

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = ASYN, BSC, CONF, DISK, ECL, ETH, HDLC, JOBS, POOL, SYS, and X25. ²
PRTTNSRPT	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QPFR	QSPDJS	PRTF	Performance printer file containing job <i>summary</i> data.
	QPFR	QSPDTD	PRTF	Performance printer file containing job state <i>transition</i> data.
	QPFR	QSPDTS	PRTF	Performance printer file containing job <i>transaction</i> data.
	QSYS	QAPMxxxx	PF	QAPMxxxx performance data collection files, where xxxx = DMPT and JOBS. ²
PRTRC	QSYS	QPSRVTRC	PRTF	Job trace printer output file. ¹
PRTRCRPT	user-lib	QTRTJOB	PF	Performance data collection <i>input</i> file containing batch job trace data.
	QPFR	QAPTDDS	PF	Performance data DDS source file.
	QSYS	QAPMDMPT	PF	Performance data collection input file containing system trace data. ²
QRYDOCLIB	QSYS	QAOSIQDL	PF	Query document library output file.
QRYDST	QSYS	QAOSILIN	PF	Incoming distribution output file.
	QSYS	QAOSILOT	PF	Outgoing distribution output file.
RCLSTG	QSYS	QPRCLDMP	PRTF	Reclaim dump output listing.
RCVDST	QSYS	QAOSIRCV	PF	Receive incoming mail distribution model database file. ²
RCVTIEF	QSYS	QPTIRCV	PRTF	Received files summary printer file. ¹
RLSDSTQ	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.
RMVDSTQ	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.
	QUSRSYS	QASNADSR	PF	SNADS routing table.
RMVDSTRTE	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.
	QUSRSYS	QASNADSR	PF	SNADS routing table.
RMVDSTSYSN	QUSRSYS	QASNADSA	PF	SNADS secondary node ID table.
RMVNETJOB	QUSRSYS	QANFNJE	PF	Network job entry database file.
RMVSOCE	QUSRSYS	QAALSOC	PF	Sphere of control file.
RMVTAPCTG	QUSRSYS	QATAMID	PF	Cartridge ID database file.
	QUSRSYS	QLTAMID	LF	Cartridge ID logical database file.
	QUSRSYS	QATACGY	PF	Category database file.
	QUSRSYS	QLTACGY	LF	Category logical file.
RMVTCPHTE	QUSRSYS	QATOCHOST	PF	TCP/IP host file.
RMVTCPIFC	QUSRSYS	QATOCIFC	PF	TCP/IP interface file.
RMVTCPPORT	QUSRSYS	QATOCPORT	PF	TCP/IP port restrictions file.
RMVTCPRSI	QUSRSYS	QATOCRSI	PF	TCP/IP remote system information file.
RMVTCPRTE	QUSRSYS	QATOCRTE	PF	TCP/IP route file.
RMVXTIDXE	QUSRSYS	QABBADMTB	PF	OfficeVision text search services administration table.
RNMTCPHTE	QUSRSYS	QATOCHOST	PF	TCP/IP host file.
RSTCFG	QSYS	QASRRSTO	PF	Model output file for configuration. ²
	QSYS	QPSRLDSP	PRTF	Restored objects status printer file. ¹
	QSYS	QSYSTAP	TAPF	Tape device file used for input.
RSTDLO	QSYS	QAOJRSTO	PF	Model output file for restored document library objects. ²

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QPRSTDLO	PRTF	Printer file for restored documents and folders. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for input.
	QSYS	QSYSTAP	TAPF	Tape device file used for input.
RSTLIB	QSYS	QASRRSTO	PF	Model output file for libraries. ²
	QSYS	QPSRLDSP	PRTF	Restored objects status printer file. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for input.
	QSYS	QSYSTAP	TAPF	Tape device file used for input.
RSTLICPGM	QSYS	QPSRLDSP	PRTF	Restored objects status printer file. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for input.
	QSYS	QSYSTAP	TAPF	Tape device file used for input.
RSTOBJ	QSYS	QASRRSTO	PF	Model output file for restored objects. ²
	QSYS	QPSRLDSP	PRTF	Restored objects status printer file. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for input.
	QSYS	QSYSTAP	TAPF	Tape device file used for input.
RSTUSRPRF	QSYS	QASRRSTO	PF	Model output file for user profiles. ²
RSTUSRPRF	QSYS	QSYSTAP	TAPF	Tape device file used for input.
RTVDOC	QSYS	QAOSIRTV	PF	Retrieve document from document library output file.
RTVDSKINF	QSYS	QAEZDISK	PF	Model file for disk information.
	QUSRSYS	QAEZDISK	PF	Database output file for disk information.
RUNQRY	QSYS	QPQUPRFL	PRTF	Printer file used for query output. ¹
SAVCFG	QSYS	QASAVOBJ	PF	Model output file for saved objects. ²
	QSYS	QPSAVOBJ	PRTF	Printer file for saved objects. ¹
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SAVCHGOBJ	QSYS	QASAVOBJ	PF	Model output file for saved objects. ²
	QSYS	QPSAVOBJ	PRTF	Printer file for saved objects. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SAVDLO	QSYS	QAOJSAVO	PF	Model output file for saved documents and folders. ²
	QSYS	QPSAVDLO	PRTF	Printer file for saved documents and folders. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SAVLIB	QSYS	QASAVOBJ	PF	Model output file for saved objects. ²
	QSYS	QPSAVOBJ	PRTF	Printer file for saved objects. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SAVOBJ	QSYS	QASAVOBJ	PF	Model output file for saved objects. ²
	QSYS	QPSAVOBJ	PRTF	Printer file for saved objects. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SAVSAVFDTA	QSYS	QASAVOBJ	PF	Model output file for saved objects. ²
	QSYS	QPSAVOBJ	PRTF	Printer file for saved objects. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SAVSECDTA	QSYS	QASAVOBJ	PF	Model output file for saved objects. ²
	QSYS	QPSAVOBJ	PRTF	Printer file for saved objects. ¹
	QSYS	QSYSDKT	DKTF	Diskette device file used for output.
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SAVSYS	QSYS	QASAVOBJ	PF	Model output file for saved objects. ²
	QSYS	QPSAVOBJ	PRTF	Printer file for saved objects. ¹

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QSYSTAP	TAPF	Tape device file used for output.
SBMDKTJOB	QSYS	QKSPLDKT	DKTF	Diskette device file used for all diskette reading.
SBMFNCJOB	QSYS	QDFNDATA	DSPF	Non-ICF finance display file.
	QUSRSYS	QFNDEVTBL	PF	File containing data for device tables.
	QUSRSYS	QFNPGMTBL	PF	File containing data for program tables.
	QUSRSYS	QFNUSRTBL	PF	File containing data for user tables.
SETTAPCGY	QUSRSYS	QATACGY	PF	Category database file.
	QUSRSYS	QLTACGY	LF	Category logical file.
SNDDSTQ	QUSRSYS	QASNADSQ	PF	SNADS distribution queues table.
SNDFNCIMG	QSYS	QCRFDWNL	ICFF	ICF file used for communication with 4701 controller.
SNDPTFORD	QGPL	Qnnnnnnn	SAVF	PTF save file, where nnnnnnn is the PTF number.
	QSYS	QESPRTF	PRTF	Printer file for PTF cover letters.
SNDSRVRQS	QGPL	Qnnnnnnn	SAVF	PTF save file, where nnnnnnn is the PTF number.
	QSYS	QESPRTF	PRTF	Printer file for PTF cover letters.
	QUSRSYS	QAEDCDBPF	PF	File containing service contact data.
STRCODE	user-lib	EVFCICFF	ICFF	ICF file used for communication with workstation.
STRCPYSCN	QSYS	QASCCPY	PF	Pattern for copy screen output file.
STRDFU	QSYS	QDFUPRT	PRTF	DFU printer file.
	QSYS	QPDZDTALOG	PRTF	DFU run-time audit log.
	QSYS	QPDZDTAPRT	PRTF	DFU run-time printer data file.
STRDKTRDR	QSYS	QKSPLDKT	DKTF	Diskette device file used for all diskette <i>reading</i> .
STRDKTWTR	QSYS	QKSPLDKT	DKTF	Diskette device file used for all diskette <i>writing</i> .

Note E - STRIDXMN Command: The STRIDXMN and STRRGZIDX commands use 9 of the 10 files in OfficeVision text search services that are also used by the STRUPDIDX command. **For those 9 files (all named as QABBxxxx), see the list under the STRUPDIDX command;** the 10th and last file in that list (QABBLADN) is *not* used by STRIDXMN or STRRGZIDX.

STRIDXMN	QUSRSYS	QABBxxxx	PF	See Note E.
STRPDM	QPDA	QPUOPRTF	PRTF	Printer file for displayed lists in PDM.

CRTPFRTA Command: All of the following files for the CRTPFRTA command are physical files (PF) or logical files (LF) that are used as model files (not actual output files) to define the record formats of files created to store performance data collected by this command. All of these model files are in the QSYS library, and the files they create are in a library determined by the user (defaults to the same library specified for the *MGTCOL object - usually QPFRDATA). Each created file stores the specific type of performance data in the format defined for the specific type of data being collected.

For the files listed below, the common part of each model file's description begins here and the unique part of each description continues under the **File Usage** column:

Model performance monitor database file that defines the record format of the data collection file created to store...

CRTPFRTA	QSYS	QAPMAPPN	PF	...APPN data. ²
	QSYS	QAPMASYN	PF	...asynchronous data. ²
	QSYS	QAPMBSC	PF	...binary synchronous data. ²
	QSYS	QAPMJOBMI	PF	...job data from MI.
	QSYS	QAPMJOBOS	PF	...job data from OS/400.

Command Name	File Library	File Name	File Type	File Usage
	QSYS	QAPMJOBWT	PF	...Job wait bucket data. ²
	QSYS	QAPMJOBWTD	PF	...Job wait bucket descriptions. ²
	QSYS	QAPMJSUM	PF	...job data summarized.
	QSYS	QAPMBUS	PF	...bus counter data. ²
	QSYS	QAPMCIOP	PF	...communications controller data. ²
	QSYS	QAPMCONF	PF	...system configuration data. ²
	QSYS	QAPMDDI	PF	...DDI distributed interface data. ²
	QSYS	QAPMDIOP	PF	...storage device controller data. ²
	QSYS	QAPMDISK	PF	...disk storage (DASD) data. ²
	QSYS	QAPMDOMINO	PF	...Domino data. ²
	QSYS	QAPMECL	PF	...ECL or token-ring LAN data. ²
	QSYS	QAPMETH	PF	...Ethernet statistics data. ²
	QSYS	QAPMFRLY	PF	...frame relay data. ²
	QSYS	QAPMHDLC	PF	...HDLC and SDLC control data. ²
	QSYS	QAPMHTTPB	PF	...HTTP server (powered by Apache) base data. ²
	QSYS	QAPMHTTPD	PF	...HTTP server (powered by Apache) detail data. ²
	QSYS	QAPMIDL	PF	...ISDN data link control file entries data. ²
	QSYS	QAPMJOBL	LF	...logical view of job data from QAPMJOBMI and QAPMJOBOS. ²
	QSYS	QAPMLAPD	PF	...ISDN LAPD file entries data. ²
	QSYS	QAPMLIOP	PF	...local work station controller (WSC) data. ²
	QSYS	QAPMMIOP	PF	...multifunction controller data. ²
	QSYS	QAPMPOOLB	PF	...main storage pool data. ²
	QSYS	QAPMPOOLT	PF	...pool tuning data. ²
	QSYS	QAPMPOOLL	LF	...logical view as pool data. ²
	QSYS	QAPMPPP	PF	...PPP protocol data. ²
	QSYS	QAPMRESP	PF	...local work station response data. ²
	QSYS	QAPMSAP	PF	...token-ring LAN, Ethernet, Distributed Data Interface, and frame relay service access point. ²
	QSYS	QAPMSBSD	PF	...subsystem description data. ²
	QSYS	QAPMSNA	PF	...SNA data. ²
	QSYS	QAPMSNADS	PF	...SNADS data. ²
	QSYS	QAPMSTND	PF	...DDI station data. ²
	QSYS	QAPMSTNE	PF	...Ethernet station data. ²
	QSYS	QAPMSTNL	PF	...token-ring LAN station data. ²
	QSYS	QAPMSTNY	PF	...frame relay station data. ²
	QSYS	QAPMSYSCPU	PF	...system CPU data. ²
	QSYS	QAPMSYSL	LF	...log view CPU sys data. ²
	QSYS	QAPMSYSTEM	PF	...system performance data. ²
	QSYS	QAPMTCP	PF	...TCP system data. ²
	QSYS	QAPMTCPIFC	PF	...TCP interface data. ²
	QSYS	QAPMTSK	PF	...system internal data. ²
	QSYS	QAPMUSRTNS	PF	...User defined transaction data. ²
	QSYS	QAPMX25	PF	...X.25 communications data. ²
STRPRTEML	QSYS	QPEMPRTF	PRTF	Emulation printer file.
STRPRTWTR	QSYS	QPSPLPRT	PRTF	Printer device file used for all printer writing.
STRQMQR	QSYS	QPQXPRTF	PRTF	Printer file used by Query CPI. ¹

Command Name	File Library	File Name	File Type	File Usage
STRQST	QQALIB	QAQAxxxx00	PF	Q & A supplied model database files. ²
	QSYS	QAQA00xxxx	LF	Q & A database model files. ²
	QSYS	QAQA00xxxx	PF	Q & A database model files. ²
	QSYS	QPQAPRT	PRTF	Q & A printer file.

Note F - STRRGZIDX Command: The STRRGZIDX and STRIDXMOM commands use 9 of the 10 files in OfficeVision text search services that are also used by the STRUPDIDX command. **For those 9 files (all named as QABBxxxx), see the list under the STRUPDIDX command;** the 10th and last file in that list (QABBLADN) is *not* used by STRRGZIDX or STRIDXMOM.

STRRGZIDX	QUSRSYS	QABBxxxx	PF	See Note F.
STRSDA	QGPL	QDDSSRC	PF	DDS source default input file.
STRSEU	QGPL	QXTSRC	PF	SEU source default input file.
	QPDA	QPSUPRTF	PRTF	SEU source member printer file.
STRSST	QSYS	QPCSMPTF	PRTF	Service tools printer file. ¹
	QTY	QATYALMF	PF	Telephony database model file of alarm record formats.
	QUSRSYS	QATYSWTE	PF	Telephony database file of user-created switch entries.
	QTY	QATYCDRF	PF	Telephony database model file of alarm record formats.
	QUSRSYS	QATYSWTE	PF	Telephony database file of user-created switch entries.

STRUPDIDX Command: The STRUPDIDX command uses the following 10 files associated with OfficeVision text search services. The first 9 of these files (all but QABBLADN) are also used by the STRIDXMOM and STRRGZIDX commands.

All of the following files have a common beginning description. Therefore, the common part of each file's description *begins* here and the unique part of each description *continues* under the **File Usage** column:

OfficeVision text search services...

STRUPDIDX	QUSRSYS	QABBADMTB	PF	...administration table.
	QUSRSYS	QABBCAN	PF	...candidates file.
	QUSRSYS	QABBCTX	PF	...context index.
	QUSRSYS	QABBDEX	PF	...external document index identifiers.
	QUSRSYS	QABBDIC	PF	...dictionary.
	QUSRSYS	QABBDIX	PF	...internal document index identifiers.
	QUSRSYS	QABBDOX	PF	...document index table.
	QUSRSYS	QABBFX	PF	...fragment index.
	QUSRSYS	QABBQTB	PF	...scheduling queue.
	QUSRSYS	QABBLADN ⁴	PF	...list of indexed LADNs (library-assigned document names).
TRCCNN	QSYS	QSYSPRT	PRTF	Connection trace printer file.
⁴ This QABBLADN file is <i>not</i> used by the STRRGZIDX and STRIDXMOM commands.				
TRCCPIC	QSYS	QACMOTRC	PF	Trace CPI-Communications database model output file. ²
	QSYS	QSYSPRT	PRTF	Trace CPI-Communications printer file. ¹
TRCICF	QSYS	QAIFTRCF	PF	Trace ICF database output file.
	QSYS	QPIFTRCF	PRTF	Trace ICF printer file. ¹
TRCINT	QSYS	QPCSMPTF	PRTF	Internal trace (for concurrent service monitor) printer file.
	QSYS	QSYSDKT	DKTF	Diskette device file for output.
	QSYS	QSYSTAP	TAPF	Tape device file for output.

Command Name	File Library	File Name	File Type	File Usage
TRCJOB	QSYS	QATRCJOB	PF	Database file that defines the record format of the file created to store trace records. ²
	QSYS	QPSRVTRC	PRTF	Job trace printer output file. ¹
UPDDTA	QSYS	QPDZDTALOG	PRTF	DFU run-time audit log.
	QSYS	QPDZDTAPRT	PRTF	DFU run-time printer data file.
VFYLNKLPDA	QSYS	QSYSPRT	PRTF	Verify link supporting LPDA-2 printer file. ¹
WRKACTJOB	QSYS	QPDSPAJB	PRTF	Active jobs display printer file. ¹
WRKALR	QUSRSYS	QAALERT	PF	Alert database file.
	QSYS	QSYSPRT	PRTF	Alert printer file. ¹
WRKCFGSTS	QSYS	QSYSPRT	PRTF	Configuration status printer file. ¹
WRKCNTINF	QUSRSYS	QAEDCDBPF	PF	Database file containing contact data.
WRKDDMF	QSYS	QPWRKDDM	PRTF	Distributed data management (DDM) file attributes printer file. ¹
WRKDEVTBL	QUSRSYS	QFNDEVTBL	PF	File containing data for device tables.
WRKDIR	QSYS	QPDSRDDL	PRTF	Directory entry <i>details</i> printer file.
	QSYS	QPDSRDMS	PRTF	Directory entry <i>summary</i> printer file.
WRKDOCCVN	QUSRSYS	QAO1CRL	LF	Document conversion <i>logical</i> file for input or output.
	QUSRSYS	QAO1CVNP	PF	Document conversion <i>physical</i> file for input or output.
	QUSRSYS	QAO1DCVN	PRTF	Document conversion printer file.
WRKDPCQ	QSYS	QPDXWRKD	PRTF	Printer file for DSNX/PC queued distribution requests. ¹
WRKDSKSTS	QSYS	QPWCDSKS	PRTF	Disk status printer file. ¹
WRKDSTL	QSYS	QPDSPLDL	PRTF	Distribution list <i>details</i> printer file.
	QSYS	QPDSPLSM	PRTF	Distribution list <i>summary</i> printer file.
WRKDSTQ	QSYS	QPDSTSTS	PRTF	Distribution status printer file. ¹
	QUSRSYS	QASNADSQ	PF	SNADS destination queues table.
WRKFCT	QRJE	QPDSPFCT	PRTF	Forms control table printer file. ¹
WRKGRPPDM	QPDA	QPUOPRTF	PRTF	Printer file for displayed lists in PDM. ¹
WRKHDWRSC	QSYS	QASUPTL	PF	Hardware resources locking database file.
WRKHTTPCFG	QUSRSYS	QATMHTTPC	PF	TCP/IP HTTP file.
WRKJOB	QSYS	QPDSPJOB	PRTF	Job display printer file. ¹
WRKJOBQ	QSYS	QPRTSPLQ	PRTF	Job queue printer file (spooling queue). ¹
WRKJOBSCDE	QSYS	QSYSPRT	PRTF	Job schedule entries printer file. ¹
WRKJRNA	QSYS	QPDSPJNA	PRTF	Journal attributes printer file. ¹
WRKLIBPDM	QPDA	QPUOPRTF	PRTF	Printer file for displayed lists in PDM. ¹
WRKMBRPDM	QPDA	QPUOPRTF	PRTF	Printer file for displayed lists in PDM. ¹
WRKMSG	QSYS	QPDSPPMSG	PRTF	Printer file for message queue messages. ¹
WRKMSGD	QSYS	QPMSGD	PRTF	Message description printer file.
WRKNAMSMTP	QSYS	QATMSMTP	PF	TCP/IP SMTP personal alias table.
	QSYS	QATMSMTPA	PF	TCP/IP SMTP system alias table.
WRKNETF	QSYS	QANFDNTF	PF	Database file for display network files. ¹
	QSYS	QPNFDNTF	PRTF	Printer file for display network files. ¹

Command Name	File Library	File Name	File Type	File Usage
WRKNETJOB	QUSRSYS	QANFNJE	PF	Database file for network job entries. ¹
	QSYS	QPNFNJE	PRTF	Printer file for network job entries. ¹
WRKOBJLCK	QSYS	QPDSPOLK	PRTF	Object locks display printer file. ¹
WRKOBJPDM	QPDA	QPUOPRTF	PRTF	Printer file for displayed lists in PDM. ¹
WRKOUTQ	QSYS	QPRTSPLQ	PRTF	Output spooling queue printer file. ¹
WRKOUTQD	QSYS	QPDSPSQD	PRTF	Output queue description. ¹
WRKPARTPDM	QPDA	QPUOPRTF	PRTF	Printer file for displayed lists in PDM. ¹
WRKPGMTBL	QUSRSYS	QFNPGMTBL	PF	File containing data for program tables.

WRKPRB Command: All 8 of the QASXxxxx files shown in the QUSRSYS library for the WRKPRB command are the same subset of files that are shown in the QUSRSYS library for the DSPPRB command. For the descriptions of these files, see the DSPPRB command.

WRKPRB	QSYS	QXPRTD	PRTF	Problem log <i>detail</i> printer file.
	QSYS	QXPRTL	PRTF	Problem log <i>summary</i> printer file.
	QUSRSYS	QASXxxxx	PF	See the DSPPRB command for these 8 files in the QUSRSYS library.
WRKPRJPDM	QPDA	QPUOPRTF	PRTF	Printer file for displayed lists in PDM. ¹
WRKQRY	QSYS	QPQUPRFL	PRTF	Printer file used for query output.
WRKQST	QSYS	QPQAPRT	PRTF	Q & A printer file.
WRKRDBDIRE	QSYS	QSYSPRT	PRTF	Distributed relational database directory printer file. ¹
WRKRDR	QSYS	QPRTRDWT	PRTF	Reader display printer file. ¹
WRKRJESSN	QRJE	QPRJESTS	PRTF	Active status printer file for RJE session. ¹
WRKRPLYE	QSYS	QPRTRPYL	PRTF	System reply list printer file. ¹
WRKSBMJOB	QSYS	QPDSPSBJ	PRTF	Submitted jobs printer file. ¹
WRKSBS	QSYS	QPDSPSBS	PRTF	Subsystem display printer file. ¹
WRKSBSJOB	QSYS	QPDSPSBJ	PRTF	Subsystem jobs display printer file. ¹
WRKSHRPOOL	QSYS	QSYSPRT	PRTF	Shared storage pools printer file. ¹
WRKSOC	QUSRSYS	QAALSOC	PF	Sphere of control database file.
WRKSPLF	QSYS	QPRTSPLF	PRTF	Spooled file printer file. ¹
WRKSPLFA	QSYS	QPDSPSFA	PRTF	Spooled file attributes printer file. ¹
WRKSPTPRD	QSYS	QSYSPRT	PRTF	Supported products printer file. ¹
WRKSRVPVD	QUSRSYS	QAEDSPI	PF	Service provider information file.
WRKSRVRQS	QUSRSYS	QANSSRI	PF	Service requester file.
WRKSSND	QRJE	QPRTSSND	PRTF	Session description printer file. ¹
WRKSYSACT	user-lib	QAITMON	PF	Performance data collection file: job and Licensed Internal Code task data. ¹
	QPFR	QAPTDDS	PF	Performance data DDS source file.
WRKSYSSTS	QSYS	QPDSPSTS	PRTF	System status printer file. ¹
WRKSYSVAL	QSYS	QSYSPRT	PRTF	System values printer file. ¹
WRKTAPCTG	QUSRSYS	QATAMID	PF	Cartridge ID database file.
	QUSRSYS	QATAMID	LF	Cartridge ID logical database file.
	QUSRSYS	QATACGY	PF	Category database file.
	QUSRSYS	QLTACGY	LF	Category logical file.
	QSYS	QSYSTAP	TAPF	Tape device file for input.
WRKTCPPPT	QUSRSYS	QATOCPTP	PF	TCP/IP point-to-point profile configuration.

Command Name	File Library	File Name	File Type	File Usage
	QUSRSYS	QATOCMODEM	PF	TCP/IP point-to-point modem configuration.
WRKTIE	QSYS	QPTIRCV	PRTF	Printer file summary of files received.
WRKTRA	QSYS	QSYSPRT	PRTF	Printer file containing list of token-ring network adapters. ¹
WRKTXIDX	QUSRSYS	QABBADMTB	PF	OfficeVision text search services administration table.
WRKUSRJOB	QSYS	QPDSPSBJ	PRTF	User jobs display printer file. ¹
WRKUSRTBL	QUSRSYS	QFNUSRTBL	PF	File containing data for user tables.
WRKWTR	QSYS	QPRTRDWT	PRTF	Writer display printer file. ¹

Printing command descriptions on the server

To print the parameter and value descriptions for a command on your iSeries server, follow these instructions:

- To print help for an entire command, do either of the following:
 - From any command line, type the command name (for example, CRTUSRPRF) and press F1. The display shows general help for the command and help for each command parameter. Press F14 to print the command help.
 - On a prompt display for a given command, move the cursor to the top line and press F1. Then press F14.
- To print the help for one CL command keyword parameter, do the following:
 - From a command line, type the CL command name and press F4 to display the command prompt display. Position the cursor anywhere on the line of the keyword parameter for which you want help. Press F1 to display the help for the keyword parameter. Press F14 to print the help.

For information about how to print PDF versions of CL information, see “Print these topics” on page 2.

Related Information for CL

See the following resource for more information about using CL.

CL Programming

This book provides a wide-range discussion of programming topics, including control language programming, programming concepts, objects and libraries, message handling, and user-defined commands and menus.



Printed in U.S.A.