

CICS for MVS/ESA



System Programming Reference

Version 4 Release 1

CICS for MVS/ESA



System Programming Reference

Version 4 Release 1

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page v.

Second edition (April 1997)

This edition applies to Version 4 Release 1 of the IBM licensed program Customer Information Control System/Enterprise Systems Architecture (CICS/ESA), program number 5655-018, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Consult the latest edition of the applicable IBM system bibliography for current information on this product.

This is the second edition of the System Programming Reference for CICS/ESA 4.1. It is based on the first edition, SC33-1171-00, which is now obsolete. Changes from the first edition are marked by the '+' sign to the left of the changes. The vertical lines in the left-hand margins indicate changes made between the CICS/ESA 3.3 edition and the CICS/ESA 4.1 first edition.

The CICS/ESA 3.3 edition remains applicable and current for users of CICS/ESA 3.3.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

At the back of this publication is a page entitled "Sending your comments to IBM". If you want to make comments, but the methods described are not available to you, please address them to:

IBM United Kingdom Laboratories Limited, Information Development,
Mail Point 095, Hursley Park, Winchester, Hampshire, England, SO21 2JN.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1977, 1997. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	v	Bibliography	ix
Programming interface information	v	CICS/ESA 4.1 library	ix
Trademarks and service marks	vi	Other CICS books	x
		Books from related libraries	x
		Other publications	x
Preface	vii	Summary of changes	xi
What this book is about	vii	+ Changes for this second CICS/ESA 4.1 edition	xi
Who should read this book	vii	Changes for the first CICS/ESA 4.1 edition	xi
What you need to know to understand this book	vii	Changes for the CICS/ESA 3.3 edition	xi
How to use this book	vii	Changes for the CICS/ESA 3.2.1 edition	xii
Notes on terminology	vii		
Determining if a publication is current	vii		

Part 1. Introduction 1

System programming commands	3	+ RESP and RESP2 options	11
+ How this book is organized	3	+ Security checking	11
Command format	4	+ The QUERY SECURITY command	13
+ CICS syntax notation and conventions used in this book	4	+ Inquiry commands	13
+ Summary of format rules	5	Browsing resource definitions	13
+ Argument values	5	Starting a browse	14
+ Data-areas and data-values	5	Retrieving the next resource	14
+ Pointer arguments	7	Ending the browse	14
+ CICS-value data areas (CVDAs)	7	Browse example	14
+ CVDA examples	7	Rules for browsing	15
+ Data types	8	Exception conditions for browsing	15
+ COBOL argument values	9	SET commands	15
+ C/370 argument values	9	Creating resource definitions	16
+ PL/I argument values	9	ATTRIBUTES option	16
+ Assembler-language argument values	10	Discarding resource definitions	17
+ Argument lengths	10	+ Exit-related commands	18
+ Null values	10	+ Defining exits	18
+ Exception conditions	10	+ Exit names	18

Part 2. System commands 19

Command information layout	20	DISCARD PROGRAM	57
ACQUIRE TERMINAL	21	DISCARD TRANCLASS	57
COLLECT STATISTICS	23	DISCARD TRANSACTION	58
+ CREATE CONNECTION	27	ENABLE PROGRAM	58
+ CREATE FILE	29	EXTRACT EXIT	61
+ CREATE LSRPOOL	31	INQUIRE AUTINSTMODEL	62
+ CREATE MAPSET	33	INQUIRE AUTOINSTALL	63
+ CREATE PARTITIONSET	35	INQUIRE CONNECTION	64
+ CREATE PARTNER	37	INQUIRE DELETSHIPED	68
+ CREATE PROFILE	39	INQUIRE DSNNAME	69
+ CREATE PROGRAM	41	INQUIRE DUMPDS	71
+ CREATE SESSIONS	43	INQUIRE EXITPROGRAM	72
+ CREATE TERMINAL	45	INQUIRE FILE	74
+ CREATE TRANCLASS	47	INQUIRE IRC	79
+ CREATE TRANSACTION	49	INQUIRE JOURNALNUM	79
+ CREATE TYPETERM	51	INQUIRE MODENAME	81
DISABLE PROGRAM	53	INQUIRE MONITOR	83
DISCARD AUTINSTMODEL	55	INQUIRE NETNAME	85
DISCARD FILE	55	INQUIRE PARTNER	85
DISCARD PARTNER	56	INQUIRE PROFILE	86
DISCARD PROFILE	56	INQUIRE PROGRAM	87

INQUIRE REQID	91	SET NETNAME	169
INQUIRE STATISTICS	94	SET PROGRAM	169
INQUIRE STORAGE	96	SET STATISTICS	172
INQUIRE SYSDUMPCODE	97	SET SYSDUMPCODE	174
INQUIRE SYSTEM	99	SET SYSTEM	176
INQUIRE TASK	104	SET TASK	179
INQUIRE TASK LIST	109	SET TCLASS	180
INQUIRE TCLASS	110	SET TDQUEUE	180
INQUIRE TDQUEUE	111	SET TERMINAL	183
INQUIRE TERMINAL	114	SET TRACEDEST	189
INQUIRE TRACEDEST	124	SET TRACEFLAG	191
INQUIRE TRACEFLAG	125	SET TRACETYPE	193
INQUIRE TRACETYPE	126	SET TRANCLASS	194
INQUIRE TRANCLASS	127	SET TRANDUMPCODE	195
INQUIRE TRANDUMPCODE	128	SET TRANSACTION	197
INQUIRE TRANSACTION	130	SET VOLUME	199
INQUIRE TSQUEUE	134	SET VTAM	200
INQUIRE VOLUME	136		
INQUIRE VTAM	138	Appendix A. CICS-value data areas used by all	
PERFORM DELETSHIPED	139	commands	203
PERFORM DUMP	140	CVDAs and numeric values in alphabetic sequence	203
PERFORM RESETTIME	141	CVDAs and numeric values in numeric sequence	207
PERFORM SECURITY REBUILD	142	+ CVDA values for the DEVICE option	211
PERFORM SHUTDOWN	143		
PERFORM STATISTICS RECORD	144	Appendix B. EXEC interface block (EIB) response	
RESYNC ENTRYNAME	147	and function codes	213
SET AUTOINSTALL	148	Response codes of EXEC CICS commands	213
SET CONNECTION	150	EXEC CICS DISCARD, INQUIRE, PERFORM, and	
SET DELETSHIPED	154	SET commands	213
SET DSNAME	156	EXEC CICS DISABLE, ENABLE, and EXTRACT	
SET DUMPDS	158	EXIT commands	213
SET FILE	160	Function codes of EXEC CICS commands	213
SET IRC	163		
SET JOURNALNUM	164	Appendix C. EXEC CICS CREATE RESP2 values	217
SET MODENAME	165		
SET MONITOR	166	Index	225

Notices

The following paragraph does not apply to any country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact Laboratory Counsel, MP151, IBM United Kingdom Laboratories, Hursley Park, Winchester, Hampshire, England SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, U.S.A.

This publication contains sample programs. Permission is hereby granted to copy and store the sample programs into a data processing machine and to use the stored copies for internal study and instruction only. No permission is granted to use the sample programs for any other purpose.

Programming interface information

This book is intended to help you use the CICS system programming commands. This book primarily documents General-use Programming Interface and Associated Guidance Information provided by CICS.

General-use programming interfaces allow the customer to write programs that obtain the services of CICS.

However, this book also documents Product-sensitive Programming Interface and Associated Guidance Information provided by CICS.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of CICS. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs by an introductory statement to a chapter or section.

Trademarks and service marks

The following terms used in this publication are trademarks or service marks of IBM Corporation in the United States or other countries:

ACF/VTAM
CICS
CICSplex
CICS/MVS
COBOL/370
DB2
IBM
IMS
MVS
MVS/ESA
RACF
System/390

BookManager
CICS OS/2
CICS/ESA
CICS/VSE
C/370
Enterprise Systems Architecture/370
IBMLink
IMS/ESA
MVS/DFP
MVS/SP
RMF
VTAM

Preface

What this book is about

This book describes the CICS/ESA system programming interface; it contains **reference** information needed to prepare COBOL, C/370, PL/I, and assembler-language application programs, using CICS commands, to be executed under CICS. **Guidance** information is in the *CICS/ESA Application Programming Guide*. For information about debugging CICS applications, see the *CICS/ESA Problem Determination Guide*.

Who should read this book

This book is for system programmers who are writing applications to be invoked as transactions for administering the running CICS system.

What you need to know to understand this book

It is assumed that you are an experienced system programmer and that you are familiar with the effects of the CICS-supplied transactions. You must be able to write application programs, and understand the contents of the CICS application programming books (that is, the *CICS/ESA Application Programming Reference* manual and the *CICS/ESA Application Programming Guide*). Anything that is already documented in those two books is not duplicated here, so you need to refer to them from time to time.

How to use this book

This book contains two major sections. The first section is an introduction that describes the common features and the overall purpose of the system programming interface commands. The second section is a description of each of the commands, in alphabetic order.

Notes on terminology

- VTAM refers to ACF/VTAM.
- The term 'SP' indicates those commands that require the special translator option 'SP'. It also indicates those commands that are subject to command security checking. The SP commands are all the INQUIRE, SET, COLLECT, and PERFORM commands, together with the DISABLE PROGRAM, ENABLE PROGRAM, EXTRACT EXIT, and RESYNC ENTRYNAME commands.
- MB equals 1 048 576 bytes.
- KB equals 1024 bytes.

Determining if a publication is current

IBM regularly updates its publications with new and changed information. When first published, both hardcopy and BookManager softcopy versions of a publication are in step, but subsequent updates will probably be available in softcopy before they are available in hardcopy.

For CICS books, these softcopy updates appear regularly on the *Transaction Processing and Data Collection Kit* CD-ROM, SK2T-0730-xx. Each reissue of the collection kit is indicated by an updated order number suffix (the -xx part). For example, collection kit SK2T-0730-06 is more up-to-date than SK2T-0730-05. The collection kit is also clearly dated on the cover.

Here's how to determine if you are looking at the most current copy of a publication:

- A publication with a higher suffix number is more recent than one with a lower suffix number. For example, the publication with order number SC33-0667-02 is more recent than the publication with order number SC33-0667-01. (Note that suffix numbers are

updated as a product moves from release to release, as well as for hardcopy updates within a given release.)

- When the softcopy version of a publication is updated for a new collection kit the order number it shares with the hardcopy version does not change. Also, the date in the edition notice remains that of the original publication. To compare softcopy with hardcopy, and softcopy with softcopy (on two editions of the collection kit, for example), check the last two characters of the publication's filename. The higher the number, the more recent the publication. For example, DFHPF104 is more recent than DFHPF103. Next to the publication titles in the CD-ROM booklet and the readme files, asterisks indicate publications that are new or changed.
- Updates to the softcopy are clearly marked by revision codes (usually a “#” character) to the left of the changes.

Bibliography

CICS/ESA 4.1 library

Evaluation and planning		
<i>Release Guide</i>	GC33-1161	April 1997
<i>Migration Guide</i>	GC33-1162	April 1997
General		
<i>CICS Family: Library Guide</i>	GC33-1226	April 1995
<i>Master Index</i>	SC33-1187	October 1994
<i>User's Handbook</i>	SX33-1188	April 1997
<i>Glossary (softcopy only)</i>	GC33-1189	n/a
Administration		
<i>Installation Guide</i>	GC33-1163	April 1997
<i>System Definition Guide</i>	SC33-1164	April 1997
<i>Customization Guide</i>	SC33-1165	April 1997
<i>Resource Definition Guide</i>	SC33-1166	April 1997
<i>Operations and Utilities Guide</i>	SC33-1167	April 1997
<i>CICS-Supplied Transactions</i>	SC33-1168	April 1997
Programming		
<i>Application Programming Guide</i>	SC33-1169	October 1994
<i>Application Programming Reference</i>	SC33-1170	April 1997
<i>System Programming Reference</i>	SC33-1171	April 1997
<i>Sample Applications Guide</i>	SC33-1173	October 1994
<i>Distributed Transaction Programming Guide</i>	SC33-1174	October 1994
<i>Front End Programming Interface User's Guide</i>	SC33-1175	October 1994
Diagnosis		
<i>Problem Determination Guide</i>	SC33-1176	October 1994
<i>Messages and Codes</i>	GC33-1177	April 1997
<i>Diagnosis Handbook</i>	LX33-6093	October 1994
<i>Diagnosis Reference</i>	LY33-6082	April 1997
<i>Data Areas</i>	LY33-6083	April 1997
<i>Supplementary Data Areas</i>	LY33-6081	October 1994
<i>Closely-Connected Program Interface</i>	LY33-6084	November 1996
Communication		
<i>Intercommunication Guide</i>	SC33-1181	April 1997
<i>Server Support for CICS Clients</i>	SC33-1591	February 1996
<i>CICS Family: Inter-product Communication</i>	SC33-0824	October 1996
<i>CICS Family: Communicating from CICS on System/390</i>	SC33-1697	October 1996
Special topics		
<i>Recovery and Restart Guide</i>	SC33-1182	October 1994
<i>Performance Guide</i>	SC33-1183	October 1994
<i>CICS-IMS Database Control Guide</i>	SC33-1184	October 1994
<i>CICS-RACF Security Guide</i>	SC33-1185	October 1994
<i>Shared Data Tables Guide</i>	SC33-1186	October 1994
<i>External CICS Interface</i>	SC33-1390	April 1997
<i>CICS ONC RPC Feature for MVS/ESA Guide</i>	SC33-1119	February 1996
<i>CICS Web Interface Guide</i>	SC33-1892	November 1996

The book that you are reading was republished in hardcopy format in April 1997 to incorporate updated information previously available only in softcopy. The right-hand column in the above table indicates the latest hardcopy editions of the CICS/ESA books available in April 1997. A book with a date earlier than April 1997 remains the current edition for CICS/ESA 4.1. Note that it is possible that other books in the library will be updated after April 1997.

When a new order is placed for the CICS/ESA 4.1 product, the books shipped with that order will be the latest hardcopy editions.

The style of IBM covers changes periodically. Books in this library have more than one style of cover.

For information about the softcopy books, see “Determining if a publication is current” on page vii. The softcopy books are regularly updated to include the latest information.

Other CICS books

- *CICS Application Migration Aid Guide*, SC33-0768
- *CICS Application Programming Primer (VS COBOL II)*, SC33-0674
- *CICS/ESA Facilities and Planning Guide* for CICS/ESA Version 3 Release 3, SC33-0654
- *CICS/ESA XRF Guide* for CICS/ESA Version 3 Release 3, SC33-0661
- *CICS Family: API Structure*, SC33-1007
- *CICS Family: General Information*, GC33-0155
- *IBM CICS Transaction Affinities Utility MVS/ESA*, SC33-1159

CICS Clients

- *CICS Clients: Administration*, SC33-1436
- *CICS Family: Client/Server Programming*, SC33-1435

Books from related libraries

You may also need to refer to the following IBM publications:

MVS/Enterprise Systems Architecture (MVS/ESA)

Short Title	Version 3 book	Version 4 book
<i>Authorized Assembler Programming Reference</i>	<i>MVS/ESA System Programming Library: Application Development Macro Reference</i> , GC28-1857	<i>MVS/ESA Application Development Reference: Services for Authorized Assembler Language Programs Volume 1</i> , GC28-1647
<i>Authorized Assembler Programming Reference Volume 2</i>	—	<i>MVS/ESA Application Development Reference: Services for Authorized Assembler Language Programs Volume 2</i> , GC28-1648
<i>Authorized Assembler Programming Reference Volume 3</i>	—	<i>MVS/ESA Application Development Reference: Services for Authorized Assembler Language Programs Volume 3</i> , GC28-1649
<i>Authorized Assembler Programming Reference Volume 4</i>	—	<i>MVS/ESA Application Development Reference: Services for Authorized Assembler Language Programs Volume 4</i> , GC28-1650
<i>JCL Reference</i>	<i>MVS/ESA JCL Reference</i> , GC28-1829	<i>MVS/ESA JCL Reference</i> , GC28-1654
<i>JCL User's Guide</i>	<i>MVS/ESA JCL User's Guide</i> , GC28-1830	<i>MVS/ESA JCL User's Guide</i> , GC28-1653
<i>System Commands</i>	<i>MVS/ESA Operations: System Commands</i> , GC28-1826	<i>MVS/ESA System Commands</i> , GC28-1626

Other publications

- *VTAM Programming*, SC23-0115
- *IMS Database Administration Guide*, SC26-4281
- *IMS Messages and Codes manual*, SC26-4290
- *IMS Utilities Reference manual*, SC26-4284
- *IMS System Administration Guide*, SC26-4282
- *An Introduction to the IBM 3270 Information Display System*, GA27-2739
- *MVS/DFP System Programming Reference*, SC26-4567.
- *MVS ESA Planning: Problem Determination and Recovery*, GC28-1629

#

Summary of changes

+ Changes for this second CICS/ESA 4.1 edition

+ This book is the second edition of the *System Programming + Reference* for CICS/ESA 4.1. Changes that were made for + the first edition are still indicated by vertical bars to the left of + the changes. Changes made for this second edition are + indicated by the '+' symbol to the left of the changes. Users + of the first edition can therefore see what has changed since + that first edition was published. Softcopy versions of this + book use both these revision indicators and use the '#' + symbol to show further changes since this second hardcopy + edition of the book was published.

| Changes for the first CICS/ESA 4.1 edition

| This edition is based on the CICS/ESA 3.3 *System Programming Reference* manual, SC33-0670.

| For CICS/ESA 4.1, the following changes have been made:

- | • Transfer of the SPOOLCLOSE, SPOOLOPEN INPUT, SPOOLOPEN OUTPUT, SPOOLREAD, and SPOOLWRITE command descriptions to the *CICS/ESA Application Programming Reference* manual
- | • Removal of storage-cushion size options from INQUIRE SYSTEM and SET SYSTEM commands, and addition of new options
- | • Addition of INQUIRE STORAGE command
- | • Addition of PSDINTERVAL, PSDINTHRS, PSDINTMINS, and PSDINTSECS options to INQUIRE VTAM and SET VTAM commands
- | • Addition of INQUIRE DELETSHIPED, PERFORM DELETSHIPED, and SET DELETSHIPED commands
- | • Addition of RELATED and LOCAL options to the INQUIRE SYSDUMPCODE, SET SYSDUMPCODE, INQUIRE TRDUMPCODE, and SET TRDUMPCODE commands
- | • Addition of INQUIRE EXITPROGRAM and INQUIRE REQID commands
- | • Addition of CICSSTATUS, GMMLLENGTH, GMMTEXT, STARTUP, and STARTUPDATE options to the INQUIRE SYSTEM command
- | • Addition of ASCII, AUTOCONNECT, DATASTREAM, and POOLTERMID options to the INQUIRE TERMINAL command
- | • Addition of GMMLLENGTH and GMMTEXT options to the SET SYSTEM command
- | • Addition of CMDSEC, DTB, DTIMEOUT, DUMPING, ISOLATE, PRIORITY, PROFILE, PROGRAM, PURGEABILITY, REMOTENAME, REMOTESYSTEM,

RESSEC, ROUTING, RTIMEOUT, RUNAWAY, SCRNSIZE, STORAGECLEAR, TASKDATAKEY, TASKDATALOC, TCLASS, TRANCLASS, TRACING, TRANPRIORITY, TRPROF, and TWASIZE options to the INQUIRE TASK command

- | • Change to the operation of the PURGETYPE option of the SET TASK and SET TERMINAL commands
- | • Addition of the ATIUSERID option to the INQUIRE and SET TDQUEUE command
- | • Addition of RUNAWAY, RUNAWAYTYPE, SHUTDOWN, STORAGECLEAR, TCLASS, and TRANCLASS options to the INQUIRE TRANSACTION command
- | • Addition of RUNAWAY, RUNAWAYTYPE, SHUTDOWN, TCLASS, and TRANCLASS options to the SET TRANSACTION command
- | • Addition of DISCARD, INQUIRE, and SET TRANCLASS commands
- | • Addition of PROGAUTOINST, PROGAUTOCTLG, and PROGAUTOEXIT options to the INQUIRE and SET SYSTEM commands
- | • Addition of program manager (PG), user interface (US), transaction manager (XM), and security manager (XS) identifiers to INQUIRE and SET TRACETYPE commands
- | • Addition of FEPI, POOL, TARGET, and NODE options to the COLLECT STATISTICS command
- | • Addition of FEPI option to the PERFORM STATISTICS RECORD command
- | • Addition of CONVERSEST, FREQUENCY, FREQUENCYHRS, FREQUENCYMIN, FREQUENCYSEC, SUBSYSTEMID, SYNCPOINTST, and TIME options to the INQUIRE MONITOR command
- + • Addition of CONVERSEST, FREQUENCY, FREQUENCYHRS, FREQUENCYMIN, FREQUENCYSEC, and SYNCPOINTST options to the SET MONITOR command.

| In addition, many editorial changes have been made to | clarify the information in this manual.

Changes for the CICS/ESA 3.3 edition

The main changes for CICS/ESA 3.3 are summarized below.

The command INQUIRE TSQUEUE with the following options: LOCATION, NUMITEMS, FLENGTH, MAXITEMLEN, and MINITEMLEN, has been introduced. The following new options have been introduced:

- BACKUPTYPE for INQUIRE DSNAME
- MAXNUMRECS, TABLE for INQUIRE FILE

- EXECKEY, EXECUTIONSET, REMOTENAME, REMOTESYSTEM, TRANSID for INQUIRE PROGRAM
- CDSASIZE, CSCS, ECDSASIZE, ECSCS, ERDSASIZE, ERSCS, EUDSASIZE, EUSCS, STOREPROTECT, UDSASIZE, USCS for INQUIRE SYSTEM
- ALTPAGEHT, ALTPAGEWD, ALTPRINTER, ALTPRTCOPYST, ALTSCRNHT, ALTSCRNWD, ALTSUFFIX, APLKYBDST, APLTEXTST, AUDALARMST, BACKTRANST, COLORST, COPYST, DEFPAGEHT, DEFPAGEWD, DEFSCRNHT, DEFSCRNWD, DISCREQST, DUALCASEST, EXTENDEDSSST, FMHPARMST, FORMFEEDST, HILIGHTST, HFORMST, KATAKANAST, LIGHTPENST, MSRCONTROLST, OBFORMATST, OBOPERIDST, OUTLINEST, PAGEHT, PAGEWD, PARTITIONSST, PRINTADAPTST, PRINTER, PROGSYMBOLST, PRTCOPYST, QUERYST, RELREQST, SCRNHT, SCRNWD, SOSIST, TEXTKYBDST, TEXTPRINTST, UCTRANST, VALIDATIONST, VFORMST, for INQUIRE TERMINAL or NETNAME
- TASKDATAKEY for INQUIRE TRANSACTION
- MAXNUMRECS, TABLE for SET FILE
- EXECUTIONSET for SET PROGRAM
- CSCS, ECSCS, ERSCS, EUSCS, USCS for SET SYSTEM
- ALTPRINTER, ALTPRTCOPYST, DISCREQST, OBFORMATST, PRINTER, PRTCOPYST, RELREQST, UCTRANST for SET TERMINAL

The following new options have been introduced:

- TASKDATALOC for INQUIRE TRANSACTION
- DATALOCATION and COBOLTYPE for INQUIRE PROGRAM
- LINKEDITMODE for ENABLE PROGRAM
- ALL RESETNOW for PERFORM STATISTICS RECORD
- RECORDNOW and RESETNOW for SET STATISTICS
- LASTRESET for COLLECT STATISTICS
- CONNSTATUS for INQUIRE CONNECTION
- CONNSTATUS for SET CONNECTION
- JOURNALNUM for INQUIRE FILE
- STARTUP and XRFSTATUS for INQUIRE SYSTEM
- NEXTTRANSID, TCAMCONTROL, and USERNAME for INQUIRE TERMINAL or NETNAME
- NEXTTRANSID and TCAMCONTROL for SET TERMINAL
- PROFILE and TPROF for INQUIRE TRANSACTION.

There are changes to the language for SESSIONTYPE in INQUIRE TERMINAL and SET TERMINAL.

Changes for the CICS/ESA 3.2.1 edition

The main changes for CICS/ESA 3.2.1 are summarized below.

Changes have been made to security, and the INQUIRE and SET commands for SYSTEM and TRANSACTION.

The new EXEC CICS DISCARD command is documented.

New INQUIRE commands for the following resources have been introduced:

- AUTINSTMODEL
- PARTNER
- PROFILE

System programming commands

This book describes the CICS system programming interface (SPI) commands. These commands are for managing the CICS system and its resources, in contrast to the application programming interface (API) commands, with which you implement end-user applications. The API is described in a companion manual to this one, the *CICS/ESA Application Programming Reference* manual. A third manual, the *CICS/ESA Application Programming Guide*, contains general information that applies to both groups of commands.

SPI commands either retrieve information about the system and its resources, or modify them. They fall into three broad categories:

- + • Commands that retrieve information about a CICS resource or system element:
 - + The INQUIRE commands
 - + COLLECT STATISTICS
- + • Commands that modify the status or definition of the system or a resource, or invoke a system process:
 - + The SET commands
 - + The CREATE commands
 - + The DISCARD commands
 - + The PERFORM commands
 - + ACQUIRE TERMINAL
- + • Commands that modify or expand system execution by means of exits:
 - + DISABLE PROGRAM
 - + ENABLE PROGRAM
 - + EXTRACT EXIT
 - + RESYNC ENTRYNAME

Together, these commands provide you with a command-level equivalent to the function of the master terminal transaction (CEMT)¹ and the trace control transaction (CETR), and an alternative to the CEDA transaction for defining resources. This means that you can write transactions for administering the running CICS system. You could, for example, provide some functions of the master terminal command for a group of users without giving them authority to use CEMT.

System programming commands are supported in the same way as application programming commands. They can be used in programs written in any CICS-supported language, and they are recognized by the command interpreter (CECI),

the execution diagnostic facility (CEDF), and the CICS translator.

+ However, there are some differences between SPI and API commands:

- + • You cannot function ship SPI commands by naming a remote resource or, generally, by specifying the SYSID option. They are executed in the CICS region in which the issuing program is running. If the command specifies a remote resource (one owned by another region), CICS uses the local (partial) definition to process the request. Consequently, if you want to use or change a resource definition in a remote region, you must cause your SPI command to be executed in that region, either by transaction routing or by distributed program link. Shared temporary storage queues, where the DFHTST TYPE=REMOTE entry is provided, are an exception.
- + • Additional security checking is available for SPI commands, as explained on page 4.
- + • Programs containing SPI commands must be translated with the SP translator option, as explained in "Security checking" on page 11.

+ There are also special considerations that apply to certain groups of commands. These notes begin on page 13.

+ How this book is organized

+ The next section in this chapter explains how SPI commands are written, including:

- + • Command format
- + • Syntax diagrams
- + • Argument types and values
- + • Data formats
- + • Language-specific notes

+ The third section contains general information that applies to all SPI commands:

- + • Exception conditions
- + • Response codes (RESP and RESP2)
- + • Security checking

+ Specifics for these groups of commands follow in the final section:

- + • Inquiries and browsing

¹ Users of earlier releases of CICS may be familiar with the old programmable interface to the master terminal program (DFHEMTA). Its use is still supported, though the documentation is available only in the *CICS Customization Guide* for releases prior to CICS/ESA Version 3.

+ You cannot use DPL to link to the CICS master terminal program, DFHEMTA. The addresses passed as parameters to DFHEMTA are valid only in the region that issues the EXEC CICS LINK command, which means you cannot route a DFHEMTA request to a remote CICS system. The same restriction also applies to the programmable interface to the RDO transaction, CEDA, invoked through program DFHEDAP.

- + • SET commands
- + • CREATE commands
- + • DISCARD commands
- + • Exit-related commands

+ You should read this material even if you already know how to write CICS commands. It begins on page 10.

+ Part 2 of this book describes the SPI commands individually, in alphabetical order. Descriptions begin with a brief statement of what the command does, followed by a syntax diagram and general information about usage. Command options are described next, also in alphabetical order. A list of the exception conditions that can arise during execution of the command follows and, in some cases, there are also examples of usage.

Command format

SPI commands are written in the same way as API commands. They begin with the words **EXECUTE CICS** (usually abbreviated **EXEC CICS**), followed by the command name, a verb or verb-and-option combination such as:

- + INQUIRE FILE
- + PERFORM SHUTDOWN
- + SET SYSTEM

+ Options that indicate details of what you want to do follow the command name. The order of the options is unimportant except when the first one is part of the command name (the FILE in INQUIRE FILE, for example).

+ SPI commands are translated into the language of the program by the same CICS translator that converts API commands, and you can mix the two categories of commands in the same program. However, you must specify the translator option SP when SPI commands are present, or the translator will not recognize them. This feature allows an installation to limit use of the SPI at compile time. Other security features restrict its use at execution time; these are described in “Security checking” on page 11.

+ The EXEC CICS that begins a command tells the translator when to begin translating. In high-level languages, you must also tell the translator when to stop, by putting a terminator at the end of the command. In COBOL, the terminator is **END-EXEC**. In C/370 and PL/I, it is a semi-colon. You do not need one in assembler, because the translator assumes

+ that the command ends on the current line unless a continuation character is present. So a command that looks like this in assembler:

```
+ EXEC CICS SET FILE(TAXPGM) OPEN
```

+ becomes

```
+ EXEC CICS SET FILE(TAXPGM) OPEN END-EXEC
```

+ in COBOL, and

```
+ EXEC CICS SET FILE(TAXPGM) OPEN;
```

+ in C/370 or PL/I.

+ For more information about translating the commands, see the *CICS/ESA Application Programming Guide* for translator options, and the *CICS/ESA System Definition Guide* for the job control language.

+ CICS syntax notation and conventions used in this book

+ In the CICS books, CICS commands are presented in a standard way.

+ Each command has a syntax diagram to show you what options there are.

+ The diagram tells you what you can put between the EXEC CICS that begins a command and the terminator that ends it. It summarizes what you can do with the particular command, and indicates relationships between different options and, sometimes, different values of an option.

+ **Note:** The diagrams and some of the examples omit the initial EXEC CICS and the language-dependent terminator, even though you must use them in your code. The diagrams also omit options that you can use in any command:

- + NOHANDLE
- + RESP
- + RESP2
- + SYSEIB

+ which have the same meaning in SPI commands as in API commands. (See the *CICS/ESA Application Programming Guide* for basic information about these options, and “Exception conditions” on page 10 for additional SPI details.)

+ You interpret the syntax diagram by following the arrows from left to right. The conventions are:

Symbol	Action
	A set of alternatives—one of which you must code.
	A set of alternatives—one of which you must code. You may code more than one of them, in any sequence.
	A set of alternatives—one of which you may code.
	A set of alternatives — any number (including none) of which you may code once, in any sequence.
	Alternatives where A is the default.
	Use with the named section in place of its name.
Punctuation and uppercase characters	Code exactly as shown.
Lowercase characters	Code your own text, as appropriate (for example, name).

+ Figure 1 shows an example. It indicates that INQUIRE + STORAGE requires you to specify either the ADDRESS + option or the NUMELEMENTS option (but not both). If, and + only if, you choose ADDRESS, you can specify ELEMENT, + FLENGTH, both, or neither. If you choose NUMELEMENTS, + you can specify ELEMENTLIST, LENGTHLIST, or TASK in + any combination (including none).

+ A list of the exception conditions that can occur on the + command appears at the bottom of the diagram. In this + case, the possibilities are the NOTAUTH and TASKIDERR + conditions.

+ Summary of format rules

+ Here is a summary of the format rules for coding CICS + commands:

- + • Follow the conventions of the language in which you are coding for general format (the column in which the command starts, the columns available to it, embedded comments, embedded blanks, and so on).
- + **Note:** The translator is not sensitive to blanks between option names and option values or the parentheses that separate them, so you can use them or not, as you wish, even in assembler.
- + • Start your command with **EXEC CICS** or **EXECUTE CICS** and end it with the terminator required by the program language (see “Command format” on page 4).

- + • If a command does not fit on a line, or you wish to break it into multiple lines, use the conventions of the language. In assembler, use a continuation character on all but the last line.
- + • Select the options you wish to use from the syntax diagram, observing the rules expressed in the diagram and the option text for required options and consistent combinations.
- + • Code punctuation and upper case letters as shown in the diagram (you can use mixed case or lowercase for keywords shown in uppercase if you prefer).
- + • Substitute your own text for lowercase letters, following the conventions of the language in which you are coding.

+ Argument values

+ The data associated with an option is called its **argument**. + Five different types appear in the syntax diagrams:

- + • data-area
- + • data-value
- + • ptr-ref (for pointer-reference)
- + • ptr-value (pointer-value)
- + • cvda (CICS-value data area)

+ Data-areas and data-values

+ Data-areas and data-values are the basic argument types. + The difference between them is the direction in which + information flows when a task executes a command. A + **data-value** is always, and exclusively, a *sender*: it conveys + data to CICS that CICS uses to process the command. A + **data-area** is a *receiver*, CICS uses it to return information to + the caller. For example, in the command:

```
+ EXEC CICS INQUIRE PROGRAM (TAXPGM)
+ USECOUNT (UCNT) END-EXEC
```

+ PROGRAM is a sender option and TAXPGM is a data-value; + it tells CICS where to find the name of the program you are + inquiring about. USECOUNT is a receiver option, and UCNT + is a data-area; CICS returns the information you requested + (the use count for this program) there.

+ In general, you can use any area (variable) for a data-area, + provided that:

- + • The data type (format) is correct. The area must be long enough and, in high-level languages, the associated variable must be defined to have the correct length and internal representation. The data types that CICS uses are discussed in “Data types” on page 8.
- + • The program logic allows the value to be changed (CICS stores into data-areas).
- + • CICS re-entrancy rules allow the value to be changed. CICS loads only one copy of any given program, no matter how many tasks are using it. To prevent tasks

+ Pointer arguments

+ A pointer-reference (abbreviated **ptr-ref** in the diagrams) is a special case of a data-area. It also is a receiver field, but CICS uses it to return a pointer to the data requested, rather than the data itself. That is, CICS stores the location (address) of the data in the argument you provide.

+ A pointer-value (abbreviated **ptr-value**) is the pointer counterpart of a data-value. That is, you *send* information to CICS in a pointer-value, but you provide the address of the data (a pointer to it), rather than the data itself.

+ The rules listed for data-areas therefore apply to pointer-references, and those for data-values to pointer-values. Each language provides a type definition for pointers, and facilities for expressing address literals that can be used for pointer-values; internally, pointers are stored in fullword binary form. See the FREEMAIN command in the *CICS/ESA Application Programming Reference* for more information about the distinction between data and pointers.

+ CICS-value data areas (CVDAs)

+ A **CVDA** (CICS-value data area) is still another special case: it is an argument to which CICS has assigned a specific and limited set of meaningful values. These values are named, both to make them intuitive and easy to remember and to keep the interface between user programs and CICS symbolic, so that version and platform changes do not require program modifications.

+ Some CVDAs send information to CICS. A sender CVDA is a special case of a data-value, and the rules for data-values apply. Others return information from CICS, and you must use the rules for data-areas. If there is any question about the direction in which the information is flowing, you can tell from the verb used in the option description. *Specifies* means that you are sending information to CICS (that is, data-value rules apply); *returns* indicates that CICS will return information in the argument (data-area rules apply).

+ CICS provides the code that converts CVDA value names to the corresponding numeric representations. (Internally, CVDAs are stored as fullword binary numbers, and you must always provide a fullword binary area for options that receive CVDA values.)

+ One way to send a CVDA value is simply to name the appropriate value (the name of the option is implied in the name of the value). For example:

```
+ EXEC CICS SET PROGRAM (TAXPGM)
+           DPLSUBSET END-EXEC
```

+ sets the EXECUTIONSET option value to DPLSUBSET.
+ EXECUTIONSET determines the set of commands which the

program is allowed to use. It has two possible values:
+ DPLSUBSET, which restricts a program to the commands allowed in a program invoked by a distributed program link, and FULLAPI, which does not restrict the command set.

+ The alternative is to use the CICS-provided DFHVALUE function, which relates the internal representation to the value name. For example, this code is equivalent to the COBOL statement above:

```
+ MOVE DFHVALUE(DPLSUBSET) TO TAXAPI.
+ EXEC CICS SET PROGRAM (TAXPGM)
+           EXECUTIONSET(TAXAPI) END-EXEC.
```

+ This technique is easier to use when program logic is complex.

+ You also use DFHVALUE when your program needs to interpret a value returned as a CVDA. For example, if you needed to perform logic based on the EXECUTIONSET value, you would write something like this:

```
+ EXEC CICS INQUIRE PROGRAM (TAXPGM)
+           EXECUTIONSET (TAXAPI) END-EXEC.
+ IF TAXAPI = DFHVALUE(FULLAPI) PERFORM STND-INIT
+ ELSE PERFORM REMOTE-INIT.
```

+ “Examples” on page 60 lists all of the CVDA value names with corresponding numeric values. These are for reference only, however; you should use value names and DFHVALUE in your code, to keep it version- and platform-independent.

+ CVDA examples

+ Here are examples in all the CICS-supported languages which show the use of CVDAs and the DFHVALUE function. In each case, the code provided:

- Tests whether the file named PAYROLL is closed.
 - If so, changes the UPDATE and DELETE option values for the file to UPDATABLE and NOTDELETABLE respectively (so that records can be updated and read, but not deleted). Note that the UPDATE option is set by using the DFHVALUE function, and that the DELETE option is set by specifying the value name. These methods are equivalent; either could have been done either way.
- + The absence of other options indicates that those values are to remain unchanged. This information could also have been expressed by specifying the options with a null values, as explained in “SET commands” on page 15.
- Returns to CICS.

+ Only the code and definitions related to this part of each program are shown.

+ COBOL version

```
+ WORKING-STORAGE SECTION.  
+ 01 FILE-STATUS-INFO.  
+   02 UOPST          PIC S9(8) COMP.  
+   02 UUPD          PIC S9(8) COMP.  
+   02 INFILE        PIC X(8).  
+   . . .  
+ CICS-REQUESTS.  
+   MOVE 'PAYROLL ' TO INFILE.  
+   EXEC CICS INQUIRE FILE(INFILE)  
+       OPENSTATUS(UOPST) END-EXEC.  
+   IF UOPST = DFHVALUE(CLOSED)  
+       MOVE DFHVALUE(UPDATABLE) TO UUPD  
+       EXEC CICS SET FILE(INFILE)  
+           UPDATE(UUPD)  
+           NOTDELETABLE END-EXEC.  
+   EXEC CICS RETURN.
```

+ Assembler-language version

```
+           DFHEISTG  
+ UOPST    DS    F      *OPEN STATUS  
+ UUPD     DS    F      *UPDATE STATUS  
+ INFILE   DS    CL8    *FILE NAME  
+           . . .  
+ MVC INFILE,=CL8'PAYROLL '  
+ EXEC CICS INQUIRE FILE(INFILE)          X  
+           OPENSTATUS(UOPST)  
+ CLC UOPST,DFHVALUE(CLOSED)  
+ BNE OPENLAB  
+ MVC UUPD,DFHVALUE(UPDATABLE)  
+ EXEC CICS SET FILE(INFILE)              X  
+           UPDATE(UUPD)                  X  
+           NOTDELETABLE  
+ OPENLAB EXEC CICS RETURN
```

+ C version

```
+ #define INFILE    "PAYROLL "  
+ main()  
+ {  
+     long int uopst, /* OPENSTATUS value */  
+     long int uupd; /* UPDATE value */  
+     . . .  
+     EXEC CICS ADDRESS EIB(dfheiptr);  
+     EXEC CICS INQUIRE FILE(INFILE)  
+         OPENSTATUS(uopst);  
+     if( uopst == DFHVALUE(CLOSED) )  
+     { uupd = DFHVALUE(UPDATABLE);  
+       EXEC CICS SET FILE(INFILE)  
+           UPDATE(uupd)  
+           NOTDELETABLE; }  
+     EXEC CICS RETURN;  
+ }
```

+ Data types

+ For most arguments, CICS uses one of four data types
+ (formats):

- + • Fullword binary (four bytes)
- + • Halfword binary (two bytes)
- + • Packed decimal (variable number of bytes)
- + • Character string (variable number of bytes)

+ The first three are all used for numeric data, but they differ in
+ length and internal format; the last is for text. The names
+ used in this book are those used in assembler language.

+ Data-areas and data-values may require any of these
+ formats. The option text tells you which one to use. CVDAs
+ are always fullword binary. Pointers are also stored in this
+ form, although you generally define them explicitly as
+ pointers or addresses. There are a few exceptions to these
+ types, including the component identifier arguments in the
+ INQUIRE and SET TRACETYPE commands, which are bit
+ strings, options where the user determines the data format,
+ and options for which CICS requires a specific structure.
+ These exceptions are rare in the SPI, however, and are
+ always noted in the option description text.

+ The data types are the same regardless of the language of
+ the program issuing the command. However, the way you
+ define data of a particular type varies with the language.
+ The rules are summarized in the language sections that
+ follow, but there are other considerations unique to each
+ language. You should refer to the material for the language
+ you are using in the *CICS/ESA Application Programming*
+ *Guide* if you are not familiar with it.

+ PL/I version

```
+ DCL (UOPST,UUPD) FIXED BIN(31), /*OPEN,UPD STATUS*/  
+     INFILE CHAR(8); /*FILE NAME */  
+     . . .  
+     INFILE='PAYROLL '  
+     EXEC CICS INQUIRE FILE(INFILE)  
+         OPENSTATUS(UOPST):  
+     IF UOPST = DFHVALUE(CLOSED) THEN DO;  
+         UUPD = DFHVALUE(UPDATABLE);  
+         EXEC CICS SET FILE(INFILE)  
+             UPDATE(UUPD)  
+             NOTDELETABLE; END;  
+     EXEC CICS RETURN;
```

+ COBOL argument values

+ In COBOL, you can use any data name of the correct data type for any argument. For a data-value, you can also use a constant that can be converted to the correct type. The ADDRESS special register can be used for both pointer-references and pointer-values, and the LENGTH special register can be used for length arguments that take a data-value.

+ The table that follows indicates how to define the correct data type.

Data type	COBOL definition
Halfword binary	PIC S9(4) COMP
Fullword binary (including CVDA)	PIC S9(8) COMP
Pointer	USAGE IS POINTER
Character string (n characters long)	PIC X(n)
Packed decimal (n decimal digits)	PIC S9(n) COMP-3

+ C/370 argument values

+ In C/370, you can use any data reference of the correct data type for a data-area, data-value, or CVDA, provided the reference is to contiguous storage. In addition, for a data-value, you can use any C/370 expression that can be converted to the correct data type. The table below shows how to define the correct data type.

Data type	C/370 definition
Halfword binary	short int
Fullword binary (including CVDA)	long int
Character string (n characters long)	unsigned char[n]
Packed decimal	Not used—see note
Note: Packed decimal arguments are not supported in C/370. Whenever there is an option that takes such an argument, there are other options that convey or return the same information in a format supported by C/370.	

+ Pointer-reference and pointer-value arguments can be any C/370 pointer reference, and pointer-values can also be any C/370 expression that can be converted to an address.

+ CICS calling sequences pass arguments by reference (the MVS convention), rather than by value (the C/370 convention). Ordinarily, the translator makes the necessary adjustments, but there are some situations in which you need to prefix your argument with an ampersand (&). See the C/370 discussion in the *CICS/ESA Application Programming Guide* for details on arguments and other aspects of writing CICS programs in C/370.

+ PL/I argument values

+ In PL/I, an argument can be any PL/I data reference of the correct data type, provided the reference is to connected storage. In addition, a data-value, a pointer-value, or sender CVDA can be any PL/I expression that can be converted to the required type, including one containing built-in functions like ADDR or LENGTH. The table below shows how to define the correct data type:

Data type	PL/I definition
Halfword binary	FIXED BIN(15)
Fullword binary (including CVDA)	FIXED BIN(31)
Pointer	POINTER
Character string (n characters long)	CHAR(n)
Packed decimal (n decimal digits)	FIXED DEC(n,0)

+ PL/I requires that the data type, precision, length and alignment attributes of a variable passed in a CALL statement match those of the corresponding argument on the ENTRY statement for the called procedure. If the attributes do not match, the PL/I compiler substitutes a dummy variable for the one specified in the CALL.

+ The translator generates ENTRY statements when it translates your CICS commands to PL/I CALLs and, if there is a mismatch between the ENTRY statement specification for an argument and the variable you specify, CICS gets a dummy variable instead of yours. Although the compiler issues a warning message when it makes such a substitution, it is easy to miss the message, and the execution results are almost never what was intended. This occurs even if there is no difference in the way the compiler implements a particular attribute value.

+ The ENTRY statements that the translator generates specify data type, precision, and length, using the values shown in the table above. Therefore, to prevent the compiler from substituting dummy variables, you must specify these attributes explicitly for variables used in CICS commands unless they happen to match the defaults. (Defaults come from a DEFAULT statement if you have used one, and from the compiler defaults otherwise.)

+ In contrast, the generated ENTRY statements do *not* specify the alignment attribute, and therefore the defaults apply. This means that alignment agreement between an argument in a CICS option and the ENTRY statement occurs only if the argument has default alignment, and happens automatically if you do not override PL/I's defaults.

+ Defaults at an installation can change and, therefore, the safest policy is to specify data type, length, and precision explicitly for variables used in CICS commands, and to omit the alignment specification.

+ If you use variable-length character strings, you need to be aware of another aspect of PL/I. PL/I prefixes character strings defined as VARYING with a two-byte length field. If

+ you name such a string as a data-value, the data CICS
+ receives starts with this length prefix—usually an unintended
+ result. (The length sent to CICS is whatever you specify in
+ the associated length option or, if you omit it, the maximum
+ length for the string plus two for the length prefix.) Similarly,
+ if you name the string as a data-area, CICS stores the
+ information requested starting at the length prefix. CICS
+ does not prefix character data with length, and so this also is
+ usually unintended.

+ Assembler-language argument values

+ In assembler language, an argument calling for a data-area,
+ data-value, or CVDA can be any relocatable expression that
+ refers to data of the correct type, including register forms
+ such as 20(0,11), and forms that use the macro-replacement
+ facilities. You can use literal constants, such as =F'1' or
+ =AL2(100), for data-values and sender CVDAs, but you
+ should not use them—or any other storage that is not to be
+ modified—for receiver arguments.

+ Pointer arguments, in contrast, are conveyed through a
+ general register in CICS assembler programs and, therefore,
+ they must be absolute expressions. For a pointer-value, you
+ specify the number of the register that contains the address
+ of the data (loading the register first if it doesn't already point
+ to it). For a pointer-reference, you specify the register in
+ which CICS is to return the address of the data. For
+ example, after execution of:

```
+ EXEC CICS INQUIRE TASK LIST
+         LISTSIZE(LISTLEN)
+         SET (9)
```

+ the address of the task list is in register 9.

+ Argument lengths

+ Arguments in character form can be variable in length; the
+ USERDATA option in the ACQUIRE TERMINAL command is
+ an example. Where this occurs, CICS provides an option
+ with which you can specify the length of the data, and you
+ must do so if you are coding in C/370. In COBOL, PL/I, and
+ assembler, however, you do not ordinarily need to specify
+ this option because, if you omit it, the translator generates
+ the length option and supplies the correct value using the
+ language facilities. In COBOL, for example, if you write:

```
+ EXEC CICS ACQUIRE TERMINAL('ABCD')
+         USERDATA(LOGONMSG) END-EXEC
```

+ the translator adds the USERDATALEN option, as if you had
+ written:

```
+ EXEC CICS ACQUIRE TERMINAL('ABCD')
+         USERDATALEN(LENGTH OF LOGONMSG)
+         USERDATA(LOGONMSG) END-EXEC
```

+ Note that the translator gets the length directly from the
+ variable name, so you must use a name with the correct
+ length associated if you omit the length option.

— APAR PQ06624

added this sentence

In COBOL, VS COBOL II, PL/I and assembler language, if
the translator option NOLENGTH is used, the translator will
not default the length options.

+ Null values

+ CICS defines a **null value** for most types of data. CICS sets
+ receiver option values to the null value corresponding to the
+ data type for the option if the option does not apply in a
+ particular situation, and you can use them in sender options
+ to indicate that you want no change to an option value. (See
+ the “Inquiry commands” on page 13 and “SET commands”
+ on page 15 for more about these uses.)

+ The null value for each data type is listed below:

Data type	Null value
Character string (n characters long)	n blanks (X'40')
Halfword binary	-1 (X'FFFF')
Fullword binary	-1 (X'FFFFFFFF')
Pointer (address)	X'FF000000'
CVDA (in receiver option)	DFHVALUE(NOTAPPLIC) (-1 or X'FFFFFFFF')
CVDA (in sender option)	DFHVALUE(IGNORE) (-1 or X'FFFFFFFF')

+ Exception conditions

+ CICS does not distinguish between SPI and API commands
+ in the flow of control after it processes a command. You
+ should read the material on this subject in the *CICS/ESA*
+ *Application Programming Guide* if you are not familiar with it,
+ because the information that follows is only a summary.

+ A program that issues a CICS command regains control at
+ the point following the command if any of the following are
+ true:

- + • The command executes normally
- + • You specify NOHANDLE or RESP in the command (you
+ can specify these options in *any* command)
- + • An exception occurs for which an IGNORE CONDITION
+ command has been issued

+ If an exception occurs for which a HANDLE CONDITION
+ command is active, control goes to the point specified in the
+ HANDLE command. Otherwise, CICS takes its default action
+ for the exception. Except where specifically noted, this
+ action is an abend. The abend codes associated with each
+ exception condition are listed in Appendix B, “EXEC

+ interface block (EIB) response and function codes” on
+ page 213.

+ RESP and RESP2 options

+ CICS sets a primary and sometimes a secondary response
+ code when it completes a command, and provides options
+ for you to inspect them. The primary code returned by the
+ RESP option is the general result—either NORMAL, meaning
+ that the command executed successfully, or the name of an
+ exception condition such as NOTAUTH (not authorized) or
+ INVREQ (invalid request). The secondary code, returned by
+ RESP2, provides a finer level of detail.

+ RESP values are similar to CVDAs in that there is a limited
+ set of values, each of which is named, and CICS translates
+ the value name to its numeric equivalent. “CVDA values for
+ the DEVICE option” on page 211 lists the correspondence,
+ but you should use the value names in your code to keep it
+ version- and platform-independent.

+ For example, here is code from a program that initializes for
+ an application. It opens and enables a file, and then checks
+ to ensure that the operation was successful before
+ continuing:

```
+ EXEC CICS SET FILE ('TAXFILE ') OPEN ENABLED  
+       RESP(RC) END-EXEC.  
+ IF RC = DFHVALUE(NORMAL) PERFORM MAIN-RTN  
+ ELSE IF RC = DFHVALUE(NOTAUTH)  
+       PERFORM SECURITY-WARNING  
+       ELSE PERFORM ERR-RTN.
```

+ Many exception conditions can have multiple causes. If you
+ need to know the exact cause, you use the RESP2 option,
+ which you can specify whenever you have specified RESP.
+ For example, if you wanted to distinguish a failure because
+ the file was remote from other failures in the example above,
+ you could add the RESP2 option to the SET FILE statement:

```
+ EXEC CICS SET FILE ('TAXFILE ') OPEN ENABLED  
+       RESP(RC) RESP2(RC2) END-EXEC
```

+ and then test explicitly for a remote file:

```
+ IF RC2 = 1 . . .
```

+ RESP2 values are numeric and predefined by CICS, like
+ RESP values, but they are not named; you use the numeric
+ values, as shown in the example. They are unique for a
+ specific command, and the RESP2 value implies the RESP
+ value, so that you do not need to test both. They are not
+ unique across commands, however, as RESP values are.
+ Both are fullword binary values, defined in the same way as
+ a CVDA in the same language:

```
+ COBOL          PIC S9(8) COMP  
+ C/370          long int  
+ PL/I           FIXED BIN(31)  
+ Assembler     F
```

+ Security checking

+ CICS uses an external security manager, such as RACF, to
+ perform security checking. The facilities CICS provides are
+ summarized in this section, but you will need to refer to other
+ manuals for full information. The *CICS/ESA CICS-RACF*
+ *Security Guide* is the most comprehensive reference; it
+ describes in detail how CICS uses RACF facilities to
+ implement security. System and resource definition options
+ that govern security are described in the *CICS/ESA System*
+ *Definition Guide* and the *CICS/ESA Resource Definition*
+ *Guide*, respectively, and the *CICS/ESA Customization Guide*
+ contains general information on CICS security.

+ Four types of security checks govern whether a particular
+ SPI command is executed:

- + Transaction
- + Command
- + Surrogate
- + Resource

+ CICS performs these checks if, and only if, the SEC system
+ initialization parameter has a value of YES.

+ The **transaction** check occurs first, at task attach time, when
+ CICS ensures that the user initiating the task has authority to
+ use the transaction that is to be executed. This check is
+ governed by the XTRAN system initialization parameter as
+ well as SEC; it is skipped if the XTRAN value is NO. The
+ remaining checks occur as necessary when commands are
+ issued.

+ **Command** checking verifies that the user is authorized to
+ use SPI commands. It is governed by the XCMD and
+ CMDSEC system initialization parameters, and the CMDSEC
+ option in the definition of the TRANSACTION being
+ executed, and occurs if the XCMD value is not NO and either
+ the CMDSEC value for the TRANSACTION is YES or for the
+ SIT is ALWAYS. If the user is not authorized, CICS rejects
+ the command with a RESP value of NOTAUTH and a
+ RESP2 value of 100. SPI commands are the ones described
+ in this book, and are the same ones that require the SP
+ option for translation (see page 4).

+ If the command associates a user to a resource, a **surrogate**
+ check may follow the command check. This check ensures
+ that the user of the task issuing the command has authority
+ to act as a surrogate for the user named in the command. It
+ occurs only if the XUSER system initialization parameter is
+ YES, and applies only to these command-option
+ combinations:

- + SET TDQUEUE with ATIUSERID
- + CREATE CONNECTION with SECURITYNAME
- + CREATE SESSIONS with USERID
- + CREATE TERMINAL with USERID

+ CICS returns a RESP2 value of 102 if the check fails.
+ (Additional checks on the assigned user occur on SET
+ TDQUEUE ATIUSERID, as detailed in the description of that
+ command.)

+ The final check is the **resource** check, to verify that the user
+ has authority to use the resource in the way required by the
+ command. Resource checking is controlled by the RESSEC
+ option in the TRANSACTION being executed, the RESSEC
+ system initialization parameter, and a system initialization
+ parameter specific to the resource type:

- + XDCT for transient data queues
- + XFCT for files
- + XJCT for journals
- + XPCT for transactions
- + XPPT for programs, map sets, partition sets, exits
- + XTST for temporary storage queues

+ Resource checking occurs only if the applicable
+ resource-type system initialization system initialization
+ parameter has a value other than NO and either the
+ RESSEC option in the TRANSACTION is YES or the
+ RESSEC system initialization parameter is ALWAYS. For
+ commands other than INQUIRE NEXT, CICS rejects the
+ command with the NOTAUTH condition and a RESP2 value
+ of 101 if a resource check fails. During a browse, however,
+ CICS simply skips resources that would fail the resource
+ check on an ordinary INQUIRE (see "Rules for browsing" on
+ page 15 for details).

+ The resources that can be protected in this way, and the SPI
+ commands that require access authority, are shown in the
+ table that follows. The letter in parentheses after the
+ command indicates whether the user needs read (R), update
+ (U) or alter (A) authority to the resource.

Resource (system initialization parameter)	Commands
Exits (XPPT option)	DISABLE PROGRAM (U) ENABLE PROGRAM (U) EXTRACT EXIT (R) INQUIRE EXITPROGRAM (R)
Files (XFCT option)	COLLECT STATISTICS FILE (R) CREATE FILE (A) DISCARD FILE (A) INQUIRE FILE (R) SET FILE (U)
Journals (XJCT option) COLLECT STATISTICS JOURNALNUM (R) INQUIRE JOURNALNUM (R) SET JOURNALNUM (U)	
Programs Map sets Partition sets (XPPT option)	COLLECT STATISTICS PROGRAM (R) CREATE MAPSET (A) CREATE PARTITIONSET (A) CREATE PROGRAM (A) DISCARD PROGRAM (A) INQUIRE PROGRAM (R) SET PROGRAM (U)
Temporary storage queues (XTST option)	INQUIRE TSQUEUE (R)
Transactions (XPCT option)	COLLECT STATISTICS TRANSACTION (R) CREATE TRANSACTION (A) DISCARD TRANSACTION (A) INQUIRE TRANSACTION (R) INQUIRE REQID TRANSID (R) SET TRANSACTION (U)
Transaction classes (XPCT option)	COLLECT STATISTICS TCLASS (R) COLLECT STATISTICS TRANCLASS (R) CREATE TRANCLASS (A) DISCARD TRANCLASS (A) INQUIRE TCLASS (R) INQUIRE TRANCLASS (R) SET TCLASS (U) SET TRANCLASS (U)
Transient data queues (XDCT option)	COLLECT STATISTICS TDQUEUE (R) DISCARD TDQUEUE (A) INQUIRE TDQUEUE (R) SET TDQUEUE (U)

+ The QUERY SECURITY command

+ You can find out whether you are authorized to access a resource or to issue a system programming command by issuing the QUERY SECURITY command. This is not an SPI command and does not access any resources, and so never raises a NOTAUTH condition. It is described in the + *CICS/ESA Application Programming Reference* manual.

+ Inquiry commands

+ The system programming commands allow you to inquire about the definition and status of most of the resources defined to CICS, and about many elements of the CICS system as well. The resources about which you can inquire are:

- + Autoinstall terminal models (AUTINSTMODEL)
- + Connections (CONNECTION)
- + Exits (EXITPROGRAM)
- + External data sets (DSNAME)
- + Files (FILE)
- + Journals (JOURNALNUM)
- + Log streams (STREAMNAME)
- + Map sets (PROGRAM)
- + Partition sets (PROGRAM)
- + Partners (PARTNER)
- + Profiles (PROFILE)
- + Programs (PROGRAM)
- + Session groups (MODENAME)
- + System dump codes (SYSDUMPCODE)
- + Temporary storage queues (TSQUEUE)
- + Terminals (TERMINAL, NETNAME)
- + Transaction dump codes (TRANDUMPCODE)
- + Transactions (TRANSACTION)
- + Transaction classes (TCLASS, TRANCLASS)
- + Transient data queues (TDQUEUE)

+ For most resource types, the options in the INQUIRE command correspond to specific elements in the definition of that resource. Such options usually have the same or similar names in the INQUIRE command and in the resource definition. Where they do not, the option text in this manual notes the corresponding definition option. Consequently, if you need additional information about the meaning of an option value, it is often helpful to refer to the definition of the resource in the *CICS/ESA Resource Definition Guide*.

+ The system elements about which you can inquire are:

- + Autoinstall for terminals (AUTOINSTALL)
- + Dump data sets (DUMPDS)
- + Interregion communication (IRC)
- + Monitor (MONITOR)
- + Requests (REQID)
- + Shipped terminal status (DELETSHIPPED)
- + Statistics (STATISTICS)
- + Storage (STORAGE)

- + System status (SYSTEM)
- + Tasks (TASK, TASK LIST)
- + Tracing (TRACEDEST, TRACEFLAG, TRACETYPE)
- + VTAM

+ Most of these elements correspond to system initialization parameters. If you need more information about them, see the system initialization table (SIT) discussion in the + *CICS/ESA System Definition Guide*.

+ Certain considerations apply to all of the inquiry commands, which are principally the INQUIREs, but also include COLLECT STATISTICS and EXTRACT EXIT.

- + • **Exception conditions:** CICS returns no information when an exception condition occurs; data-areas named in receiver options are unchanged.
- + • **Exclusive control:** A task inquiring about a resource, system setting, or system component does not get exclusive control of the object of the inquiry. The information returned may be changed by another task or system event at any time.
- # The resource currently being inquired on should not be deleted since the current resource is used to position to the next resource on a subsequent GETNEXT command. Only after the subsequent GETNEXT command can the resource be deleted, since it is no longer required for positioning within this browse request.
- + • **Browsing:** Resources defined in the first (resource) list can be retrieved sequentially, as explained in “Browsing resource definitions.”
- + • **Inapplicable options:** If you specify a receiver option that does not apply to the resource about which you are inquiring, CICS generally returns the appropriate “null value,” as defined in “Null values” on page 10. (In a few cases, an exception is raised; these cases are noted in the command descriptions.)

+ For example, if you include BLOCKFORMAT in an INQUIRE TDQUEUE command that specifies an intrapartition transient data queue, CICS returns the value NOTAPPLIC to the CVDA you provide, because BLOCKFORMAT is valid for extrapartition queues only.

Browsing resource definitions

The INQUIRE commands that apply to resources ordinarily retrieve information about a **single** resource that you name when you issue the command, and the individual command syntax discussions in the next section describe them in this form.

However, there is another form that enables you to browse through some or all of the definitions of a given type. The resource types that you are allowed to browse are those in the first list in “Inquiry commands,” plus requests (REQIDs).

Starting a browse

A browse involves three steps. First, you issue the INQUIRE command with an additional option, START, to set up the browse. This command does not produce any information; it just tells CICS what you are going to do. The general form of the command is:

```
INQUIRE resource-type START
```

In addition to the START option, there are several differences in the way you issue this set-up command from the normal syntax:

- You identify the resource type only, without providing a resource name. That is, the resource type appears without its customary data-value.
- You omit all of the options in which CICS returns information to you.
- + • You also omit options that send information to CICS, other than the resource type. (INQUIRE EXITPROGRAM is an exception to this rule; you can limit the browse by supplying additional information on the START, as explained in the descriptions of these commands.)

Generally, CICS returns resource definitions to you in the order it keeps them internally. You cannot control this order, and you should not depend on it always being the same. For a few resource types, however, CICS returns definitions in alphabetic order of resource name. These are:

- + • Programs, map sets, and partition sets
- Temporary storage queues
- Transactions
- Transaction classes

For these resources only, you can specify a starting point for the browse with the AT option on the INQUIRE START:

```
INQUIRE resource_type START AT(data-value)
```

The AT data-value is the name at which you want to start. It must be in the correct format for a name of the resource type being browsed, but it does not have to correspond to an installed resource; it is used only to start the browse at the proper point in the resource list. CICS will restrict the definitions that it returns on your INQUIRE NEXT commands to resources with names equal to or greater (in the collating sequence) than the value you provide.

Retrieving the next resource

In the second step of a browse, you issue the INQUIRE command repetitively with another new option, NEXT. CICS + returns one resource definition for each INQUIRE NEXT. The general format is:

```
INQUIRE resource_type(data-area) NEXT option...option
```

Apart from the addition of NEXT, the options are almost the same on an INQUIRE NEXT as on a single INQUIRE for the same type of resource. Again, however, there are some differences:

- Instead of specifying the name of the resource (a data-value), you provide a **data-area** of the same length for CICS to return the name of the next resource to you.
- Options by which CICS returns data to you are used in the same way as on the single-resource form.
- + • A few options, such as the CONNECTION option on INQUIRE MODENAME, change their roles in a browse. These differences also are noted in the commands to which they apply.

You repeat the INQUIRE NEXT command until you have seen the resource definitions you want or have exhausted the definitions. After you have retrieved the last of them, CICS raises the END condition on subsequent INQUIRE NEXTs, leaving any data-areas you provided unchanged. However, you do not have to retrieve all the definitions; you can stop the browse at any time.

Ending the browse

Stopping the browse is the final step. To do so you issue an INQUIRE for the resource type with just the END option, thus:

```
INQUIRE resource_type END
```

Browse example

Here is an example of a typical browse sequence. This code retrieves the names of all the files installed in the system and calls a subroutine to process information about the recovery characteristics if the file is open.

```

EXEC CICS INQUIRE FILE START END-EXEC.
PERFORM UNTIL RESP CODE = DFHRESP(END)
  EXEC CICS INQUIRE FILE(FILENAME) NEXT
  OPENSTATUS(OPENSTAT)
  RECOVSTAT(RCVRSTAT)
  FWDRECSTATUS(FWDSTAT)
  RESP(RESPCODE) END-EXEC
+ IF RESP CODE = DFHRESP(NORMAL)
+   IF OPENSTAT = DFHVALUE(OPEN)
+     CALL RCVY-RTN USING RCVRSTAT FWDSTAT
+   END-IF
+ ELSE CALL ERROR-RTN END-IF
END-PERFORM.
EXEC CICS INQUIRE FILE END END-EXEC.

```

Rules for browsing

In addition to the syntax changes described above, there are some rules you should note about browsing resource definitions:

1. Your position in a browse is associated with your task, so that it is preserved across LINK and XCTL commands.

Note: Programs that run as part of a program list table (PLT) during CICS initialization or termination run under a single task. Consequently, they should terminate explicitly any browse they begin, in order not to conflict with other programs in the same PLT.
2. A task can browse more than one type of resource at the same time, but can have only one browse in progress for a particular resource type.
3. A SYNCPOINT command does not end a browse or affect your position in it.
4. Resource definitions are not locked during a browse, and another task may change the definitions while you are inquiring on them.
5. Nonetheless, you should always end a resource browse explicitly, rather than allowing end-of-task processing to do so implicitly, because a browse holds control blocks that other tasks may require for browsing.
6. INQUIRE NEXT commands usually do not cause a task switch. Therefore, a task browsing a long list of resources may exceed the runaway task interval without ceding control, causing CICS to abend it with an AICA code. If this occurs, you need to intersperse a SUSPEND command periodically among your INQUIRE NEXTs.
- + 7. During a browse in a task for which resource security checking is in effect, CICS returns only those definitions that the user is authorized to see. The others are skipped without any indication.

Exception conditions for browsing

Two conditions can occur on the browse forms of an INQUIRE command, in addition to those that apply to the single-resource form of the command:

END

RESP2 values:

- 2 INQUIRE NEXT has been issued, but there are no more resource definitions of the type being browsed.

ILLOGIC

RESP2 values:

- 1 A START has been given when a browse of the same resource type is already in progress, or a NEXT or an END has been given without a preceding START.

SET commands

You can change most of the system elements and resource definitions about which you can inquire, although in general you cannot change as many option values as you can retrieve. Changes are made with a SET command naming the resource or system element.

Like the INQUIRE commands, SET commands follow some general rules:

- **Exceptions:** When a SET command results in an exception condition, CICS makes as few of the requested changes as possible. To establish which, if any, changes have been made, you can issue the corresponding INQUIRE command.
- **Permanence:** If you change a system setting or resource definition element that is ordinarily recorded in the CICS global catalog, the change is also recorded in the catalog and thus preserved over a warm or emergency restart. If the information is not ordinarily recorded, it lasts only for the current execution of CICS. In a cold or initial start, the catalog information is discarded and all effects of earlier SET commands are lost.
- **Recoverability:** SET commands are not recoverable. Their effects are not backed out if the task that issued them abends or issues a SYNCPOINT ROLLBACK command. Consequently, SET commands do not lock resources, and you do not need to precede a SET with the corresponding INQUIRE command.
- **“No change” values:** Except where there is a default value for an option, CICS does not change the value associated with an option that you omit. However, there is a second way to indicate that you want no change. If you specify the null value in a sender option that is not required, CICS leaves the option value unchanged. Although you can get the same effect by omitting the option if there is no default, the ability to specify a “no

change” value allows you to vary the options in a command as well as the option values, simplifying your code in some situations.

For example, suppose you needed to change many different combinations of options, depending on the outcome of some calculations. Your code might look something like this:

```
IF ... MOVE DFHVALUE(NOTDELETABLE) TO DEL
ELSE MOVE DFHVALUE(IGNORE) TO DEL.
IF ... MOVE 2 TO POOL
ELSE MOVE -1 TO POOL.
IF ... MOVE 'TAXID.MASTER' to DSN
ELSE MOVE SPACES TO DSN.
EXEC CICS SET FILE('TAXMSTR ') DELETE(DEL)
        LSRPOOLID(POOL) DSNNAME(DSN) END-EXEC.
```

See “Null values” on page 10 for more about null values.

Note: There are a few options, such as the NEXTTRANSID option in a SET TERMINAL command, for which blanks (the null value for a character field) are a meaningful value. For these options, there is no null value, and you must omit the option if you do not want to change its value; these cases are noted in the option descriptions.

Creating resource definitions

CREATE commands allow you to add resource definitions to the local CICS region by program, so that you can write applications to administer a running CICS system. These definitions are equivalent to those produced by CEDA transactions. They are recorded in the CICS global catalog and persist over a warm or emergency restart.

However, CREATE commands neither refer to nor record in the CICS system definition (CSD) file. Consequently, the resulting definitions are lost on a cold or initial start, and you cannot refer to them in a CEDA transaction.

You can create definitions for the following types of resources:

- Connections
- Files
- LSR pools
- Map sets
- Partition sets
- Partners
- Profiles
- Programs
- Sessions
- Terminals
- Terminal types (TYPETERMs)
- Transaction classes
- Transactions

A CREATE command corresponds to a combined CEDA DEFINE and INSTALL, except for not updating the CSD file.

If there is no resource of the same name and type already installed, the new definition is added to the resources of your CICS region. (Definitions always apply to the local CICS region, even if they describe resources located on a remote system.) If the resource was already installed, the new definition replaces the old one, and an implicit discard of the old resource occurs as well. In this case, most restrictions that would apply to a DISCARD command naming the same resource, apply to the CREATE.

During the processing, CICS syncpoints your task, as if a SYNCPOINT command had been issued along with the CREATE. Changes made to recoverable resources between the CREATE and task start (or the most recent syncpoint) are committed if processing is successful and rolled back if not. (For TERMINAL definitions and CONNECTION-SESSIONS definitions that require more than one CREATE command to complete, the syncpoint takes place on the final CREATE of the sequence.)

If an error is detected before installation processing begins, installation is not attempted. CICS raises an exception condition and returns control to the issuing task without syncpointing. However, some errors are detected later in the process and cause rollback, and all successful CREATEs cause a commit. Tasks using these commands need to be written with these commit effects in mind.

In addition, the implied syncpoint means that CREATE commands cannot be issued in a program invoked by a distributed program link unless the LINK command specifies SYNCONRETURN, in a program with an EXECUTIONSET value of DPLSUBSET, or in any other situation where syncpoint is not allowed.

CREATE commands can be executed at any time after the start of the third phase of CICS initialization. This means they can be used in programs specified in the second section of the program load table for postinitialization (PLTPI) as well as during normal CICS execution.

ATTRIBUTES option

The specifics of the resource definition that a CREATE command installs are conveyed through the ATTRIBUTES option value, which is a character string listing the attributes of the resource. You specify attributes and attribute values in text form, in the same way that you do on a CEDA DEFINE screen. This character string is analyzed at the time the CREATE command is executed, and consequently must consist entirely of text, rather than variable names, in a single string. The syntax within the string is provided for each CREATE command, using the same conventions as command syntax, except for the attribute values as noted below. However, the contents are *not* parsed by the translator, which checks only the command syntax, shown in the main diagram.

Attribute values appear essentially as they do on CEDA DEFINE screens. However, because DEFINE screens are

preformatted and ATTRIBUTES strings are not, you need to know the following rules:

- Attributes may appear in any order (you do not have to follow the order in the syntax diagram or in the CEDA command).
- The name of an attribute must be that shown in the syntax diagram or the abbreviation permitted in the corresponding CEDA DEFINE entry (see the discussion of DEFINE in the *CICS/ESA Resource Definition Guide*).

Note: Abbreviations can change from release to release, and thus full spellings are safest.

- The attribute string is not converted to uppercase, in contrast to inputs to CEDA and the DFHCSDUP utility. Attribute names are recognized regardless whether you use upper, lower or mixed case, as are value names assigned by CICS (those shown in uppercase letters in the syntax diagram). However, other character values—resource names and message text, for example—are taken as is, so that you need to supply them in the intended case.
- The argument value, if any, also must follow the rules for the same attribute in a CEDA DEFINE panel. Where there are a limited number of possible values, they are listed in the attributes diagram in uppercase. Otherwise the diagram indicates only the form of the value, using the following conventions:

char*n* A character string of length *n* or, where the argument can be of variable length, of maximum length *n*.

hex*n* A string of hexadecimal characters of length *n* or, where the argument can be of variable length, of maximum length *n*.

n1-n2 A number in the range *n1* to *n2*.

Note: You can omit trailing blanks in character arguments, trailing X'00's in hexadecimal arguments, and leading zeros in numeric arguments. In all cases, you should refer to the *CICS/ESA Resource Definition Guide* for specific rules about the argument values.

- You can separate attributes by one or more blanks for readability, but a blank is required only between an attribute that has no argument and the next attribute. Commas and other separators are not allowed. Blanks may also appear between an attribute name and the parentheses that surround its argument, and between the parentheses and the argument value, but they are not necessary. Thus both of these, and obvious combinations, are correct:

```
ATTRIBUTES ( 'UCTRAN (NO)RTIMEOUT (10 )' )
ATTRIBUTES( ' UCTRAN(NO) RTIMEOUT( 10 ) ' )
```

- No quote marks are required within the attribute string (you need them around the whole string if you use a literal, of course, as in the example above). If you want quotes within your text—in the DESCRIPTION attribute, for example—use two quote characters for each one that you want to appear in the text, as you do in literal constants that contain quotes.

- Very few attributes require specification, and omitting one is equivalent to not keying a value for it on a CEDA screen. Where the default value is always the same, it is shown in the diagram in the same way as in syntax diagrams. However, some defaults depend on the values of other attributes, and these are not shown. (You cannot define your own defaults, because CREATEs do not use the CSD file.)
- For some resource types, you can default all attributes. If you wish to do this, set the length of the string to zero in the ATTRLEN option. You must still specify the ATTRIBUTES option in this case, even though the data-value you provide is not examined.
- The ATTRLEN option can be omitted when it is not zero if it is the length of the variable specified in ATTRIBUTES and you are not coding in C/370, as explained in “Argument lengths” on page 10.

If you make an error in the ATTRIBUTES string, CICS raises the INVREQ condition with an appropriate RESP2 value. Appendix C, “EXEC CICS CREATE RESP2 values” on page 217 lists the RESP2 values that apply.

Discarding resource definitions

The DISCARD command deletes the definition of a resource + installed in the local CICS system, so that the system no + longer has access to the resource, or makes a model + ineligible for use as a model. It reverses the effect of the + installation of the resource, which can occur at system + startup, through a subsequent CREATE command or CEDA + transaction, or by an automatic installation process.

Each DISCARD command removes the definition of one resource. You can remove definitions for the following types of resources:

- Autoinstall models for terminals
- + Connections
- Files
- + Journals
- Partners
- Profiles
- Programs, map sets, and partition sets
- + Terminals
- Transaction classes
- Transactions
- + Transient data queues

You cannot discard a resource that is currently in use. For example, you cannot discard a PROFILE definition if some installed TRANSACTION definition still points to it, or a FILE that is open, or a TRANSACTION that is scheduled for execution.

In addition, some resources are not eligible for discard at all. These include resources whose names begin with the letters DFH (reserved for CICS-supplied definitions), and

transactions whose names begin with C (also reserved for CICS).

+ Some DISCARD commands cause a syncpoint on behalf of the issuing task, as the CREATE commands do. For these commands, the discussion of syncpoint considerations on page 16 applies.

DISCARD commands are recorded in the CICS catalog, so that their effects persist over a warm or emergency restart, but they do not modify the CSD file and thus are lost on a cold or initial start.

+ Exit-related commands

+ In CICS, an **exit** is installation-supplied code that is invoked either at specific, CICS-defined points within CICS system code, or by an application request that uses the exit mechanism (such as a DB2 or IMS request). There are two types: global user exits and task-related user exits. Global user exits are always invoked at CICS-defined points in system code; task-related exits can be invoked both ways. The *CICS/ESA Customization Guide* lists the points in CICS code at which global exits may be invoked, describes how and when task-related exits are driven, and gives full details for programming exits.

+ Five SPI commands are related to exits:

- + ENABLE PROGRAM
- + DISABLE PROGRAM
- + EXTRACT EXIT
- + RESYNC ENTRYNAME
- + INQUIRE EXITPROGRAM

+ You can use them in any language supported by CICS, even though the exit itself must be coded in assembler.

+ Defining exits

+ The only way to define an exit in CICS—that is, to install it so that the code gets executed—is to issue the ENABLE PROGRAM command.

+ Similarly, the only way to delete the definition is to issue the corresponding DISABLE PROGRAM EXITALL command or shut down the system. Exit definitions last only for the current execution of CICS. They are not recorded in keypoints, the CICS global catalog, or the CSD file, and therefore do not survive a shutdown of any kind.

+ ENABLE and DISABLE PROGRAM commands affect only the CICS region in which they are issued. Even if CICS

+ system code or exit program code is shared among several executing CICS regions, the exit must be defined and deleted separately in each region that uses it.

+ Moreover, these commands are not recoverable; their effects are not backed out if the task that issued them fails or issues a SYNCPOINT ROLLBACK command.

+ Exit names

+ The code that an exit executes is contained in one or more ordinary load modules (a module may be used both by an exit and a user transaction, in fact). You identify the first module to be executed in an exit by naming it in the PROGRAM option of the ENABLE PROGRAM command that creates the exit. The exit can execute other modules as well, but you tell CICS where to start, just as you name only the first program to be executed in a TRANSACTION definition.

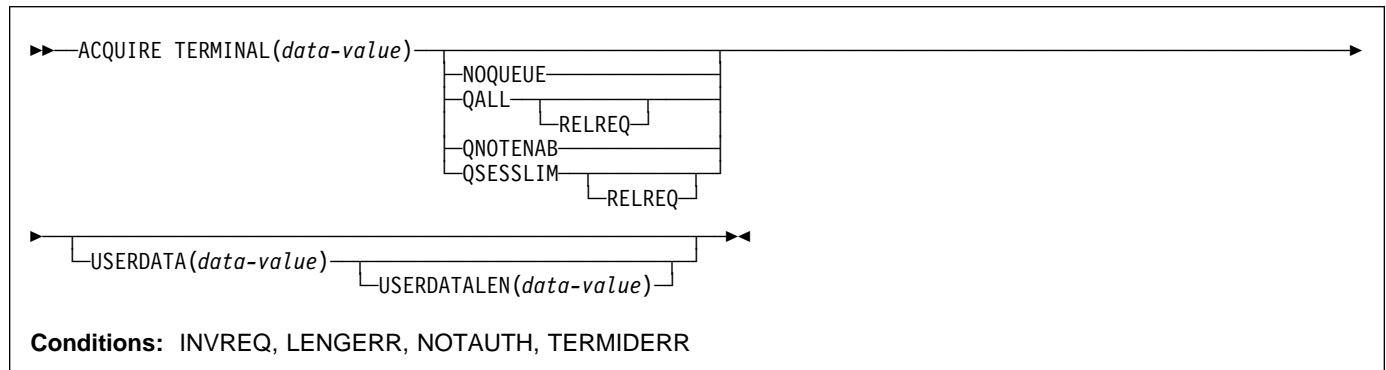
+ Exits are named by the ENTRYNAME value in the initial ENABLE PROGRAM command, not the PROGRAM value, although you can omit the ENTRYNAME option and allow its value to *default* to the PROGRAM value. Exit names must be unique, however, and if a program is used first by more than one exit, only one of them can be named by default in this way. Moreover, even when an exit and its first program have the same name, they are separate entities of different types.

+ Because of this default (and some history), CICS requires that you always identify an exit in the same way that you did in the ENABLE PROGRAM command that created it—that is, by coding (or omitting) the same PROGRAM and ENTRYNAME values. RESYNC ENTRYNAME is an exception; you specify the exit name in the ENTRYNAME option, regardless of whether you used ENTRYNAME or PROGRAM to assign the name initially. Also, in the INQUIRE EXITPROGRAM command, the option that names the initial program is EXITPROGRAM rather than PROGRAM.

+ Like modules invoked by user transactions, load modules used by exits must be defined as PROGRAM resources, either explicitly or by autoinstallation, and they must have an ENABLESTATUS value of ENABLED at the time of invocation. In addition, the initial program for an exit must be in ENABLED status at the time of the ENABLE PROGRAM command that creates the exit. However, the ENABLESTATUS of a program is independent of any exits that use it, and it is not affected by ENABLE and DISABLE PROGRAM commands that refer to it.

ACQUIRE TERMINAL

Acquire a session with a terminal.



Description

The ACQUIRE TERMINAL command enables you to tell CICS to acquire a session with a particular terminal.

The terminal you specify must be a VTAM terminal, and it cannot be an APPC, LU6.1, or IRC session. It must already be defined to CICS, either in an installed TERMINAL definition or by the autoinstall process, and it must be local to the system on which the ACQUIRE TERMINAL is issued, not remote.

This means that, if the terminal was autoinstalled, you must issue the ACQUIRE command before CICS deletes the terminal definition.

CICS normally deletes an autoinstalled terminal definition if the session ends and is not re-established within the interval specified in the AIRDELAY value in the system initialization table. The terminal does not have to be reacquired within this interval, however; after you issue the command, CICS suspends its time-out and does not delete the definition while waiting for the session to be re-established.

CICS processes an ACQUIRE command by sending a SIMLOGON request to VTAM (the queueing options on the command are for VTAM use and correspond to those on a SIMLOGON request). The task that issued the command is dispatchable as soon as this occurs. It is not notified of the eventual result of the VTAM request, nor when the terminal is actually acquired, and the terminal does not become associated with the task.

The request is sent straight to VTAM unless the terminal is already in session with the requesting CICS system. If it is, and NOQUEUE or QNOTENAB are present, CICS rejects the request as invalid (because a SIMLOGON would fail under these circumstances). Otherwise, CICS stores the request until the terminal's current session ends and then sends it to VTAM. For this reason, requests may be queued by VTAM in a different order from the order in which they were originally issued.

After it has been issued, an ACQUIRE TERMINAL request cannot be canceled, and you cannot ordinarily determine whether an ACQUIRE TERMINAL has been issued for a particular terminal.

Options

NOQUEUE

specifies that VTAM should not queue the request. Consequently, the ACQUIRE succeeds only if the terminal is immediately available.

QALL

specifies that VTAM should queue the request if the terminal is not enabled for sessions or is at its session limit (that is, in session with another VTAM application).

QNOTENAB

specifies that VTAM should queue the request only if the terminal is not enabled for sessions.

QSESSLIM

specifies that VTAM should queue the request only if the terminal is at its session limit (that is, in session with another VTAM application).

RELREQ

is meaningful only if the QALL or QSESSLIM option is set. The RELREQ option specifies that, if the requested terminal is already in session with another VTAM application, that application is notified of your request via its RELREQ exit routine. If RELREQ is not specified, the other application is not notified.

If the other application is a CICS system, the RELREQ value of the terminal definition in that system determines whether the request to release the terminal will be honored. RELREQ is specified on the TYPETERM definition associated with the terminal.

TERMINAL(data-value)

is the 4-character identifier of the terminal with which CICS is to acquire a session.

ACQUIRE TERMINAL

| **USERDATA**(*data-value*)

| specifies the data area containing the logon user data, if
| any. VTAM simulates a logon when CICS asks to
| acquire a terminal. This data corresponds to user data
| that sometimes accompanies a real logon. VTAM
| passes it to the application (in this case, the requesting
| CICS system) when the terminal has been acquired
| successfully. See the description of the EXTRACT
| LOGON command in the *CICS/ESA Application
| Programming Reference* for programming information.

| **USERDATALEN**(*data-value*)

| specifies the length, as a halfword binary value, of the
| user data. Because of a VTAM limitation, the maximum
| length of the user data is restricted to 255 bytes.

| **Conditions**

| **INVREQ**

| RESP2 values:

- | **2** The terminal is a remote terminal.
- | **3** The terminal is LU6.1, APPC, IRC or a non-VTAM
| device.
- | **4** The terminal is not in service; that is, it is not
| available for use.
- | **5** VTAM is not open.
- + **7** CICS is already in the process of acquiring this
+ session.
- | **8** NOQUEUE and QNOTENAB options are invalid
| for a logged-on device.

| **LENGERR**

| RESP2 value:

- | **6** Out-of-range value supplied in the
| USERDATALEN option.

| **NOTAUTH**

| RESP2 value:

- | **100** The user associated with the issuing task is not
| authorized to use this command.

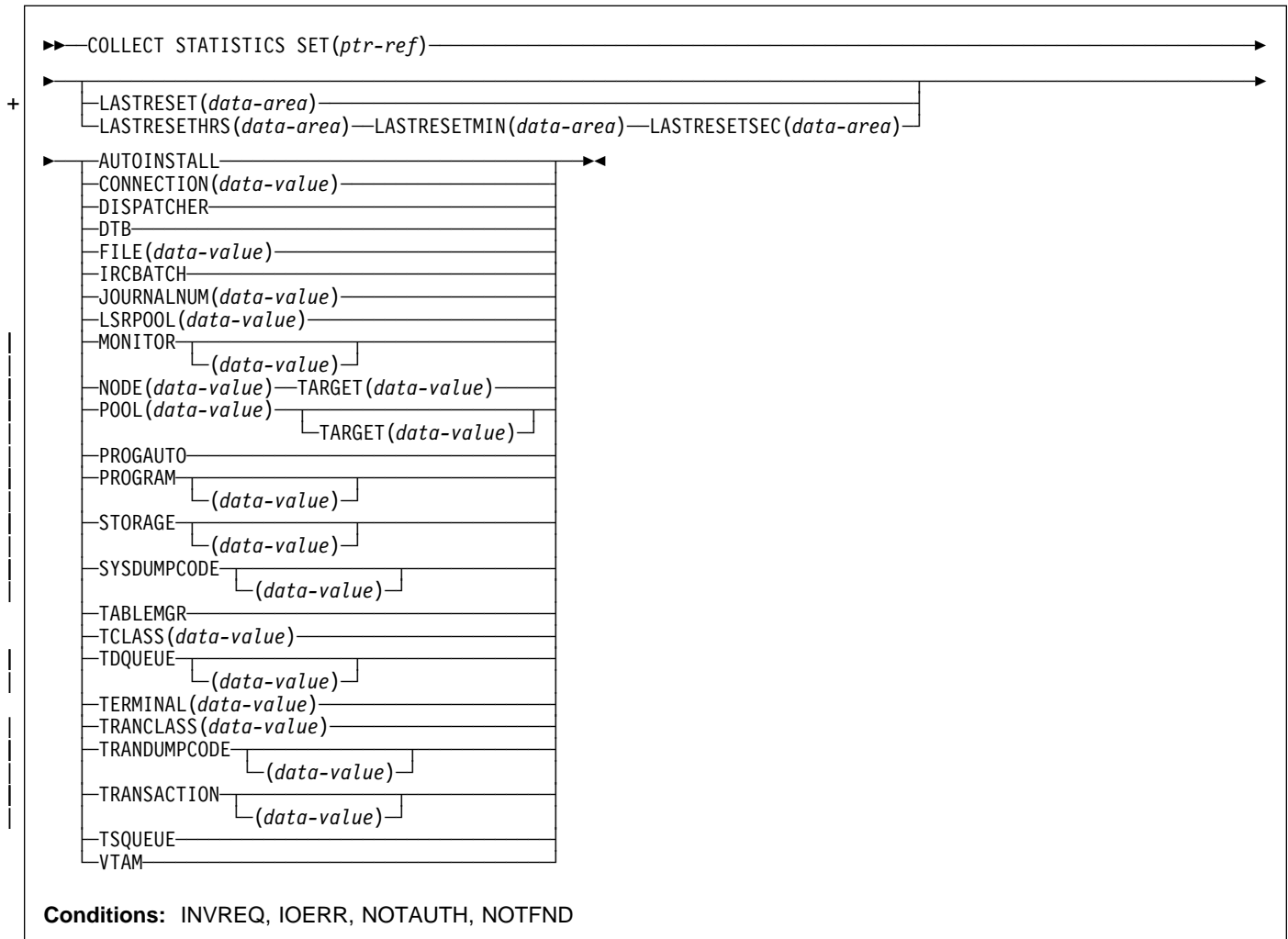
| **TERMIDERR**

| RESP2 value:

- | **1** The terminal cannot be found.

COLLECT STATISTICS

Retrieve the current statistics for a single resource or global statistics for a class of resources.



Description

The COLLECT STATISTICS command returns to the invoking application the current statistics for a particular resource, or overall statistics for the resources of a given type. For example, you can get the statistics for global transaction activity in your CICS system (such as the total number of transactions attached), or you can specify a single transaction that you are interested in (such as CEMT).

The statistics that CICS gives you are those that have been accumulated after the expiry of the last statistics collection interval, end-of-day expiry, or requested reset. (Statistics already written to the SMF data set cannot be accessed.) The COLLECT STATISTICS command does not cause the statistics counters to be reset.

CICS obtains enough storage below 16MB for the data returned from this command, and returns a pointer to this area. The first two bytes of the area contain its length. This storage can be reused by subsequent COLLECT

STATISTICS commands, so you should store elsewhere any data that is required beyond the next issue of the command. CICS releases this storage at task termination.

This section contains Product-sensitive Programming Interface information.

Not all resource types provide both global and specific statistics. Table 1 on page 24 tells you which statistics are available for each resource type, and gives the copybook name for each set of available statistics. The copybooks define the format of the returned statistics. Where no copybook name is given in the global statistics column, global statistics are not available for the resource type; similarly, where there is no entry in the specific statistics column, you cannot get statistics for an individual resource.

COLLECT STATISTICS

Table 1. Statistics and resource types

Resource type	Global statistics	Specific statistics
AUTOINSTALL	DFHA04DS	-
CONNECTION	-	DFHA14DS
DISPATCHER	DFHDSGDS	-
DTB	DFHA05DS	-
FEPI CONNECTION	-	DFHA23DS
FEPI POOL	-	DFHA22DS
FEPI TARGET	-	DFHA24DS
FILE	-	DFHA17DS
IRCBATCH	DFHA19DS	-
JOURNALNUM	-	DFHA13DS
LSRPOOL	-	DFHA08DS
MONITOR	DFHMNGDS	DFHMNTDS
PROGAUTO	DFHPGGDS	-
PROGRAM	DFHLDGDS	DFHLDRDS
STORAGE	DFHMSDS	DFHSMDDS
SYSDUMPCODE	DFHSDGDS	DFHSDRDS
TABLEMGR	DFHA16DS	-
TCLASS	-	DFHXMCD
TDQUEUE	DFHA11DS	DFHA10DS
TERMINAL	-	DFHA06DS
TRANCLASS	-	DFHXMCD
TRANDUMPCODE	DFHTDGDS	DFHTDRDS
TRANSACTION	DFHXMGDS	DFHXRDS
TSQUEUE	DFHA12DS	-
VTAM	DFHA03DS	-

Copybooks are provided in ASSEMBLER, COBOL, and PL/I. (There is no copybook for C.) The names of the copybooks are the same in each language. You can find them in the following libraries:

ASSEMBLER	CICS410.SDFHMAC
COBOL	CICS410.SDFHCOB
PL/I	CICS410.SDFHPL1

Note: Some of the copybooks contain packed fields. Before these fields are used, they should be checked for hexadecimal zeros. The COBOL versions of the fields have been redefined as numeric with a suffix of -R for this purpose.

For further information about these copybooks, see the *CICS/ESA Performance Guide*.

Options

AUTOINSTALL

requests global statistics on autoinstall.

CONNECTION(*data-value*)

requests statistics for a connection to a remote system or region; *data-value* is the 4-character identifier (from its CONNECTION definition) of the system or region.

DISPATCHER

requests global statistics on the dispatcher domain.

DTB

requests global statistics on the dynamic transaction backout function.

FILE(*data-value*)

requests statistics for a file; *data-value* is the 8-character identifier of the file (from its FILE definition).

IRCBATCH

requests global statistics on the IRC batch function.

JOURNALNUM(*data-value*)

requests statistics for a journal; *data-value* is the number of the journal, in half-word binary format. Journal numbers range from 1 to 99. '1' identifies the system log.

LASTRESET(*data-area*)

returns a 4-byte packed decimal field in the format Ohhmmss+, giving the time at which the counters for the requested statistics were last reset. This is usually the time of the expiry of the last interval. The last reset time is always returned in local time.

There are two formats for the reset time:

- A composite (packed decimal format Ohhmmss+), which you obtain by using the LASTRESET option.
- Separate hours, minutes, and seconds, which you obtain by specifying the LASTRESETHRS, LASTRESETMIN, and LASTRESETSEC options respectively.

LASTRESETHRS(*data-area*)

returns a fullword binary field giving the hours component of the time at which the counters for the requested statistics were last reset (see the LASTRESET option).

LASTRESETMIN(*data-area*)

returns a fullword binary field giving the minutes component of the time at which the counters for the requested statistics were last reset (see the LASTRESET option).

LASTRESETSEC(*data-area*)

returns a fullword binary field giving the seconds component of the time at which the counters for the requested statistics were last reset (see the LASTRESET option).

LSRPOOL(*data-value*)

requests statistics on a VSAM LSR pool; *data-value* is the pool number, in the range 1–8, in fullword binary form.

MONITOR(*data-value*)

requests performance class statistics for a task when a *data-value* is supplied. The *data-value* is the task number, in 4-byte packed decimal format. (For programming information, see EIBTASKN in Appendix A of the *CICS/ESA Application Programming Reference*.) Without a *data-value*, MONITOR requests global performance class statistics.

The monitoring performance class must be active for any statistics to be returned. If performance class is not active, the NOTFND condition is returned. For background information on monitoring, see the *CICS/ESA Performance Guide*.

NODE(*data-value*) **TARGET** (*data-value*)

requests statistics for a FEPI connection. The **NODE** *data-value* is the 8-character name of the terminal which FEPI simulates, and the **TARGET** *data-value* is the 8-character name of the system to which FEPI appears as a secondary logical unit.

POOL(*data-value*)

requests statistics for a FEPI pool; *data-value* is the 8-character name of the pool.

POOL(*data-value*) **TARGET**(*data-value*)

requests statistics for a FEPI target within a FEPI pool. The **POOL** *data-value* identifies the pool, and the **TARGET** *data-value* identifies the system within the pool for which statistics are requested.

PROGAUTO

requests global statistics on the autoinstalled program definitions.

PROGRAM(*data-value*)

requests statistics for a program when a *data-value* is supplied. The *data-value* is the 8-character name of the program PROGRAM definition. Without a *data-value*, PROGRAM requests the global program statistics.

SET(*ptr-ref*)

specifies a pointer reference to be set to the address of the data area containing the returned statistics. The first 2 bytes of the data area contain the length of the data area in halfword binary form.

STORAGE(*data-value*)

+ requests statistics for a storage domain subpool when a *data-value* is present. The *data-value* is the 8-character name of a storage domain subpool. A complete list of the possible subpool names is documented in the *CICS/ESA Performance Guide*. Without a *data-value*, this option requests the global statistics for the CICS dynamic storage areas.

SYSDUMPCODE(*data-value*)

requests statistics for a system dump code when a *data-value* is supplied. The *data-value* is the 8-character dump code. Without a *data-value*, SYSDUMPCODE requests global statistics on system dumps.

TABLEMGR

requests global statistics on the table manager.

TCLASS(*data-value*)

requests statistics for a transaction class; *data-value* is the class number, in the range 0–10, in fullword binary form. Transaction classes are no longer identified by number, but instead by an 8-character identifier.

When you use the TCLASS option to request statistics for a class (as opposed to TRANCLASS), a conversion from fullword binary number to 8-character value is made on your behalf (for example, TCLASS(01) becomes the equivalent of TRANCLASS('DFHTCL01')).

TDQUEUE(*data-value*)

requests statistics for a transient data queue when *data-value* is supplied. The *data-value* is the 4-character name of the queue (from the destination control table (DCT) definition). Without a *data-value*, TDQUEUE requests the global statistics for transient data.

TERMINAL(*data-value*)

requests statistics for a terminal; *data-value* is the 4-character terminal identifier (from the TERMINAL definition).

TRANCLASS(*data-value*)

requests statistics for a transaction class; *data-value* is the 8-character name of the class from the TRANCLASS definition.

TRANDUMPCODE(*data-value*)

requests statistics for a transaction dump code when a *data-value* is supplied. The *data-value* is the 4-character dump code. Without a *data-value*, TRANDUMPCODE requests global statistics on transaction dumps.

TRANSACTION(*data-value*)

requests statistics for a transaction when a *data-value* is supplied. The *data-value* is the 4-character transaction identifier (from the TRANSACTION definition). Without a *data-value*, TRANSACTION requests global statistics on transactions.

TSQUEUE

requests global statistics on temporary storage.

VTAM

requests global statistics on VTAM.

Conditions**INVREQ**

RESP2 value:

- 4 The TCLASS value was not in the range 1–10, or the LSRPOOL value was not in the range 1–8.

COLLECT STATISTICS

IOERR

RESP2 value:

- 3 The requested statistics area was not functioning. This will happen if, for instance, statistics control blocks are overwritten.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.
- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

NOTFND

RESP2 values:

- 1 The requested resource cannot be found (for example, if the PROGRAM name you specify does not exist in the system).
- 2 The type of resource is not defined in the CICS system (for example, FEPI statistics are requested with POOL or NODE when the system initialization table specifies FEPI=NO).

Examples

CICS provides a sample COLLECT STATISTICS application (DFH0STAT) that makes use of virtually all the options described in this section. This set of programs illustrates ways of using the COLLECT STATISTICS and INQUIRE commands of CICS/ESA 4.1 to produce information about a CICS/ESA system. The reports include a CICS and MVS storage analysis that can be used as an aid to specifying the DSA LIMIT parameters.

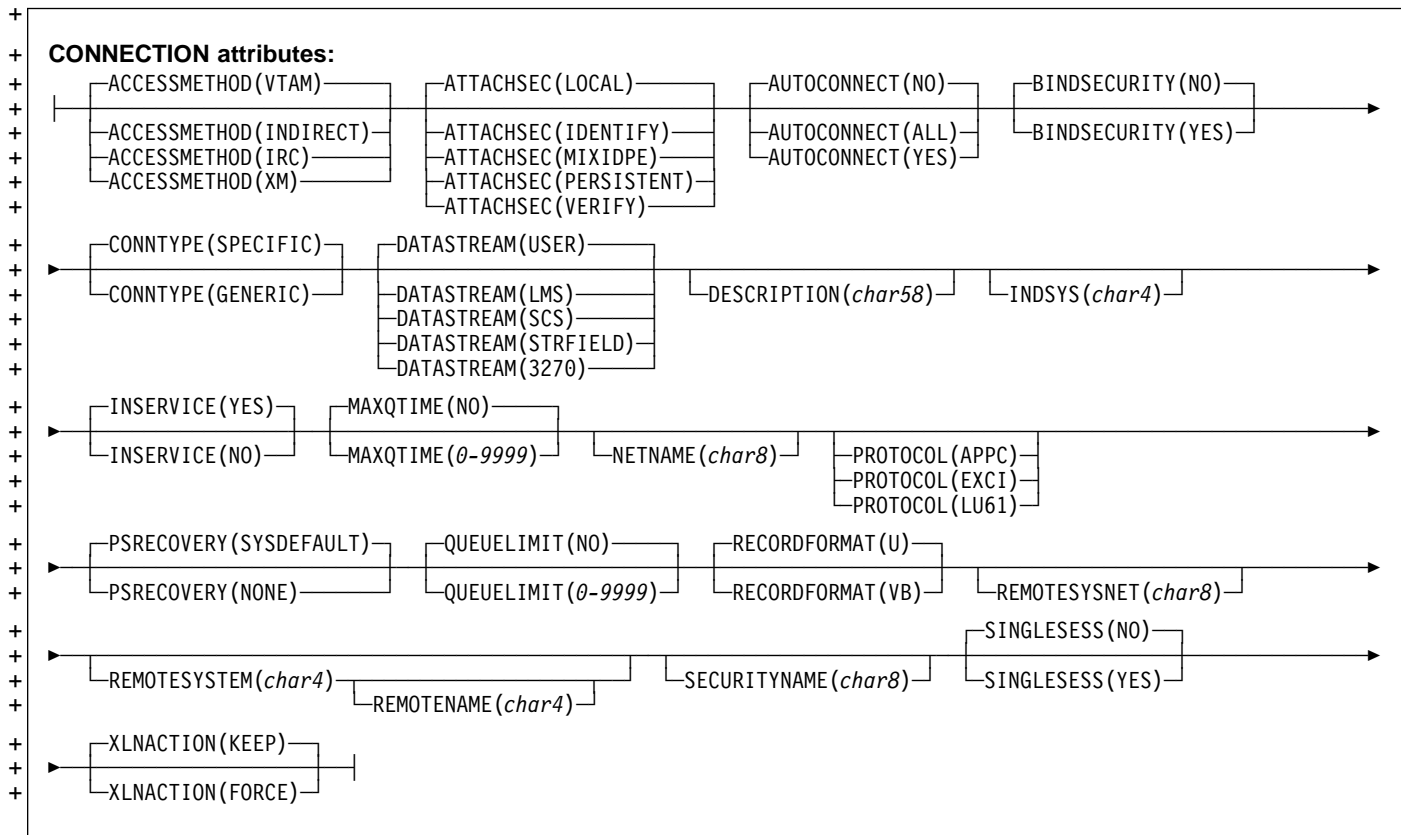
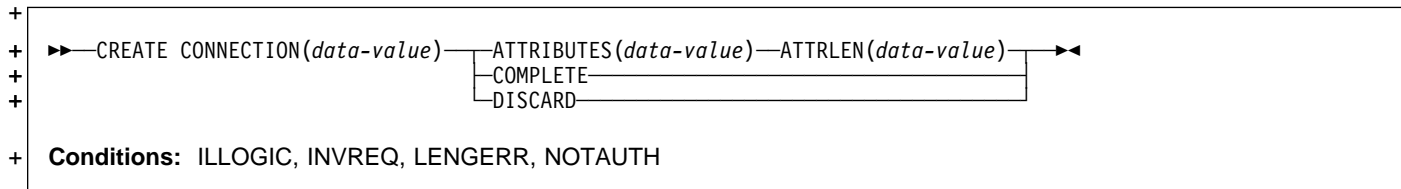
See the *CICS/ESA Performance Guide* for information on installing and operating the DFH0STAT application. The source code for the application can be found in CICS410.SDFHSAMP.

+ **CREATE CONNECTION**

+ **APAR PQ02462**
 + added EXEC CICS CREATE to CICS 4.1

+ **Function**

+ Define a CONNECTION in the local CICS region.



APAR PQ09156
 # added this note

Note to COBOL programmers: In the syntax above, you
 # must use --- ATTRIBUTES (data-area)--- instead of
 # --- ATTRIBUTES (data-value)--- .

+ **Usage**

+ CREATE CONNECTION commands, in combination with
 + CREATE SESSIONS commands, add the definition of a
 + CONNECTION and its SESSIONS to the local CICS region.
 + The definitions are built without reference to data stored on
 + the CSD file. See "Creating resource definitions" on page 16
 + for other general rules about CREATE commands.

+ To create a new CONNECTION, you issue a series of
 + commands in this order:

CREATE CONNECTION

- + • CREATE CONNECTION with the ATTRIBUTES and ATTRLEN options
 - + • CREATE SESSIONS
 - + • Additional CREATE SESSIONS if desired (only one group of sessions is required, but you can define additional groups)
 - + • CREATE CONNECTION with the COMPLETE option.
- + The CONNECTION is not added until all of these steps take place. During the time the definition is being built (that is, between the initial and final CREATE CONNECTIONs), you may not:
- + • Define other resources of any type, including other connections
 - + • Issue a SYNCPOINT (or any command that implies one)
 - + • Terminate your task (normally)

+ However, if you encounter an error or problem during the course of building a CONNECTION definition, you can terminate the process at any point by issuing a CREATE CONNECTION DISCARD command. If you do this, CICS will discard the partial CONNECTION definition and any SESSIONS created for it.

+ Otherwise, when the final CREATE CONNECTION COMPLETE command is issued, CICS adds the CONNECTION and its SESSIONS to its resource definitions, replacing a CONNECTION definition of the same name if one exists.

+ CICS also performs an implicit SYNCPOINT command during the processing of the final CREATE for a connection, unless it contains an error that can be detected early in the processing. The syncpoint commits uncommitted changes to recoverable resources made up to that point in the task if the definition is successful, and rolls back changes, as if SYNCPOINT ROLLBACK had been issued, if the definition fails or ends in a DISCARD.

+ Options

- + **ATTRIBUTES**(*data-value*)
 - + specifies the attributes of the CONNECTION being added. The list of attributes must be coded as a single character string using the syntax shown in **CONNECTION attributes**. See “ATTRIBUTES option” on page 16 for general rules for specifying attributes, and the CONNECTION chapter in the *CICS/ESA Resource Definition Guide* for details about specific attributes.
 - + **Note:** You can assign default values for all attributes of a CONNECTION definition by specifying an ATTRLEN value of 0. You still need to specify the ATTRIBUTES option, however, even though its value is not used.

- + **ATTRLEN**(*data-value*)
 - + specifies the length in bytes of the character string supplied in the ATTRIBUTES option, as a halfword binary value. The length can be from 0 to 32767.
- + **COMPLETE**
 - + specifies that the set of definitions for this CONNECTION is complete and should be added to the CICS system.

- + **CONNECTION**(*data-value*)
 - + specifies the 4-character name of the CONNECTION definition to be added.

- + **DISCARD**
 - + specifies that the CONNECTION definition under construction will not be completed and that it and any SESSIONS created for it are to be discarded and *not* added.

+ Conditions

+ ILLOGIC

- + RESP2 values:
 - + **2** The command cannot be executed because an earlier CONNECTION or TERMINAL pool definition has not yet been completed.

+ INVREQ

- + RESP2 values:
 - + **n** There is a syntactical error in the ATTRIBUTES string, or an error occurred during either the discard or resource definition phase of the processing. See Appendix C, “EXEC CICS CREATE RESP2 values” on page 217 for information on RESP2 values.
 - + **200** The command was executed in a program defined with an EXECUTIONSET value of DPLSUBSET or a program invoked from a remote system by a distributed program link without the SYNCONRETURN option.

+ LENGERR

- + RESP2 values:
 - + **1** The length specified in ATTRLEN is negative.

+ NOTAUTH

- + RESP2 values:
 - + **100** The user associated with the issuing task is not authorized to use this command.
 - + **102** The user associated with the task issuing the CREATE CONNECTION command is not an authorized surrogate of the user specified in SECURITYNAME.

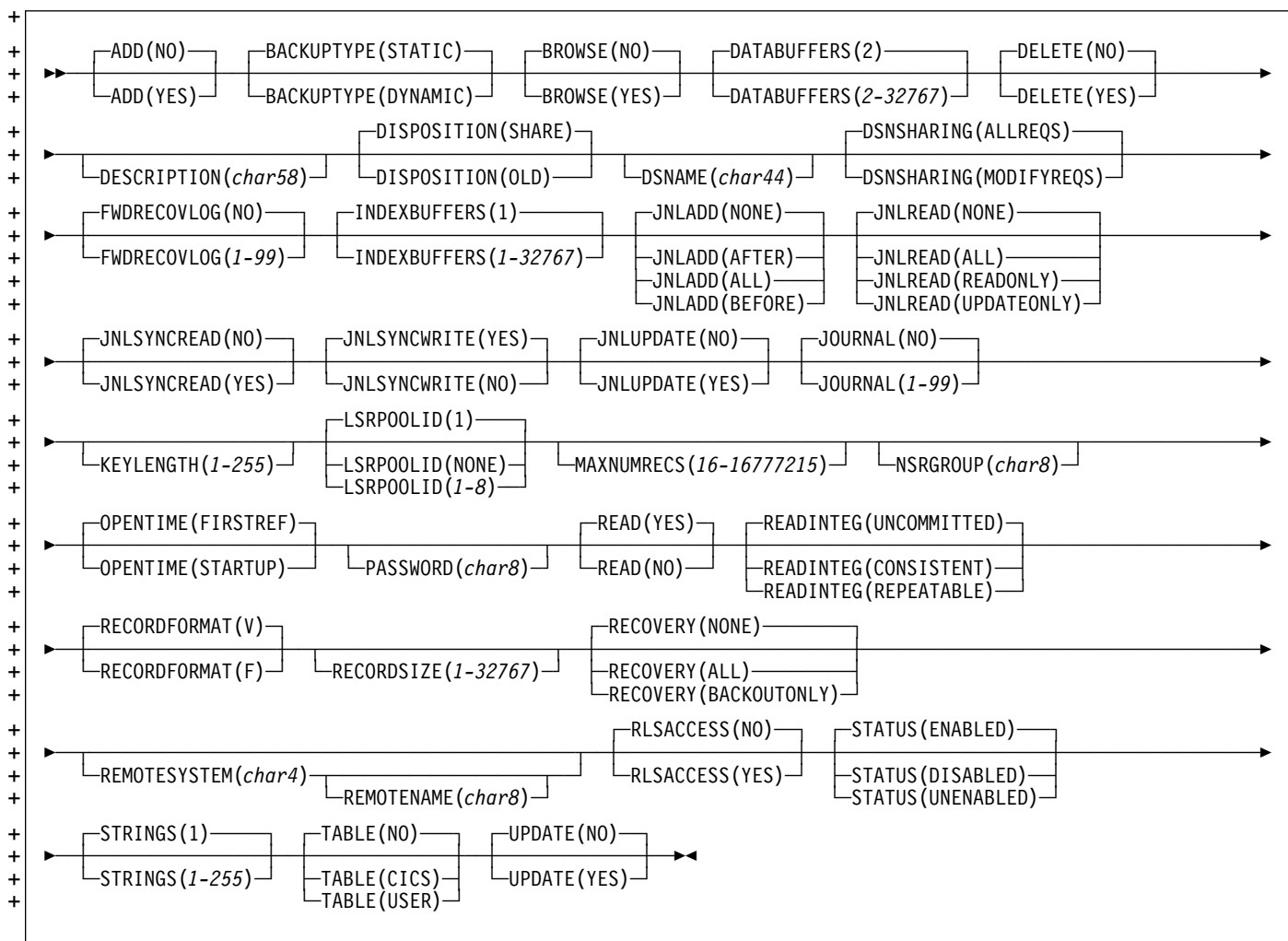
+ CREATE FILE

APAR PQ02462
 + added EXEC CICS CREATE to CICS 4.1

+ Function

+ Define a FILE in the local CICS region.

+ **▶** CREATE FILE(*data-value*)—ATTRIBUTES(*data-value*)—ATTRLEN(*data-value*)—**◀**
 + **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH



APAR PQ09156
 # added this note

Note to COBOL programmers: In the syntax above, you must use --- ATTRIBUTES (data-area)--- instead of --- ATTRIBUTES (data-value)--- .

+ Usage

+ The CREATE FILE command builds a FILE definition, + without reference to data stored on the CSD file. If there is + already a file by the name you specify in the local CICS + region, the new definition replaces the old one; if not, the + new definition is added.

CREATE FILE

- + A syncpoint is implicit in CREATE FILE processing, except
- + when an exception condition is detected early in processing
- + the command. Uncommitted changes to recoverable
- + resources made up to that point in the task are committed if
- + the CREATE executes successfully and rolled back if not.
- + See “Creating resource definitions” on page 16 for other
- + general rules governing CREATE commands.

+ Options

+ ATTRIBUTES(*data-value*)

- + specifies the attributes of the FILE being added. The list
- + of attributes must be coded as a single character string
- + using the syntax shown in **FILE attributes**. See
- + “ATTRIBUTES option” on page 16 for general rules for
- + specifying attributes, and the FILE chapter in the
- + *CICS/ESA Resource Definition Guide* for details about
- + specific attributes.

- + **Note:** You can assign default values for all attributes of
- + a FILE definition by specifying an ATTRLEN value of 0.
- + You still need to specify the ATTRIBUTES option,
- + however, even though its value is not used.

+ ATTRLEN(*data-value*)

- + specifies the length in bytes of the character string
- + supplied in the ATTRIBUTES option, as a halfword
- + binary value. The length can be from 0 to 32767.

+ FILE(*data-value*)

- + specifies the 8-character name of the FILE definition to
- + be added to the CICS region.

+ Conditions

+ ILLOGIC

- + RESP2 values:

- + **2** The command cannot be executed because an
- + earlier CONNECTION or TERMINAL pool
- + definition has not yet been completed.

+ INVREQ

- + RESP2 values:

- + **n** There is a syntactical error in the ATTRIBUTES
- + string, or an error occurred during either the
- + discard or resource definition phase of the
- + processing. See Appendix C, “EXEC CICS
- + CREATE RESP2 values” on page 217 for
- + information on RESP2 values.
- + **200** The command was executed in a program defined
- + with an EXECUTIONSET value of DPLSUBSET or
- + a program invoked from a remote system by a
- + distributed program link without the
- + SYNCONRETURN option.

+ LENGERR

- + RESP2 values:

- + **1** The length you have specified in ATTRLEN is
- + negative.

+ NOTAUTH

- + RESP2 values:

- + **100** The user associated with the issuing task is not
- + authorized to use this command.
- + **101** The user associated with the issuing task is not
- + authorized to create a FILE definition with this
- + name.

+ CREATE LSRPOOL

APAR PQ02462
 + added EXEC CICS CREATE to CICS 4.1

+ Function

+ Define an LSR pool in the local CICS region.

+ **▶**—CREATE LSRPOOL(*data-value*)—ATTRIBUTES(*data-value*)—ATTRLEN(*data-value*)—**▶**

+ **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH

LSRPOOL attributes:

DATA512(3-32767)	DATA1K(3-32767)	DATA2K(3-32767)	DATA4K(3-32767)
DATA8K(3-32767)	DATA12K(3-32767)	DATA16K(3-32767)	DATA20K(3-32767)
DATA24K(3-32767)	DATA28K(3-32767)	DATA32K(3-32767)	DESCRIPTION(char58)
HSDATA4K(0-16777215)	HSDATA8K(0-16777215)	HSDATA12K(0-16777215)	
HSDATA16K(0-16777215)	HSDATA20K(0-16777215)	HSDATA24K(0-16777215)	
HSDATA28K(0-16777215)	HSDATA32K(0-16777215)	HSINDEX4K(0-16777215)	
HSINDEX8K(0-16777215)	HSINDEX12K(0-16777215)	HSINDEX16K(0-16777215)	
HSINDEX20K(0-16777215)	HSINDEX24K(0-16777215)	HSINDEX28K(0-16777215)	
HSINDEX32K(0-16777215)	INDEX512(3-32767)	INDEX1K(3-32767)	INDEX2K(3-32767)
INDEX4K(3-32767)	INDEX8K(3-32767)	INDEX12K(3-32767)	INDEX16K(3-32767)
INDEX20K(3-32767)	INDEX24K(3-32767)	INDEX28K(3-32767)	INDEX32K(3-32767)
LSRPOOLID(1)	SHARELIMIT(50)		
LSRPOOLID(1-8)	MAXKEYLENGTH(0-255)	SHARELIMIT(1-100)	STRINGS(1-255)

APAR PQ09156

added this note

Note to COBOL programmers: In the syntax above, you
 # must use --- ATTRIBUTES (data-area)--- instead of
 # --- ATTRIBUTES (data-value)--- .

+ Usage

+ The CREATE LSRPOOL command builds the definition of a
 + VSAM local shared resources (LSR) pool, without reference
 + to data stored on the CSD file. LSR pools must have unique
 + LSRPOOLID values within a CICS region. If the local region
 + already contains a definition with the same LSRPOOLID
 + value, the new definition replaces the old one; if not, the new
 + definition is added. (Unlike most resource definitions, the
 + name you specify in the LSRPOOL option does not
 + determine replacement; instead the LSRPOOLID value
 + governs.)

CREATE LSRPOOL

+ **Note:** When you replace the definition of a pool that is currently open, the new definition does not take effect until the next time the pool is built. The pool is not rebuilt until all of the files using it are closed and one is reopened subsequently.

+ A syncpoint is implicit in CREATE LSRPOOL processing, except when an exception condition is detected early in processing the command. Uncommitted changes to recoverable resources made up to that point in the task are committed if the CREATE executes successfully and rolled back if not. See "Creating resource definitions" on page 16 for other general rules governing CREATE commands.

+ Options

+ ATTRIBUTES(*data-value*)

+ specifies the attributes of the LSR pool being added. The list of attributes must be coded as a single character string using the syntax shown in **LSRPOOL attributes**. See "ATTRIBUTES option" on page 16 for general rules for specifying attributes, and the LSRPOOL chapter in the *CICS/ESA Resource Definition Guide* for details about specific attributes.

+ **Note:** You can assign default values for all attributes of a LSRPOOL definition by specifying an ATTRLEN value of 0. You still need to specify the ATTRIBUTES option, however, even though its value is not used.

+ ATTRLEN(*data-value*)

+ specifies the length in bytes of the character string supplied in the ATTRIBUTES option, as a halfword binary value. The length can be from 0 to 32767.

+ LSRPOOL(*data-value*)

+ specifies the 8-character name of the LSRPOOL definition to be added to the CICS region.

+ Conditions

+ ILLOGIC

+ RESP2 values:

+ **2** The command cannot be executed because an earlier CONNECTION or TERMINAL pool definition has not yet been completed.

+ INVREQ

+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES string, or an error occurred during either the discard or resource definition phase of the processing. See Appendix C, "EXEC CICS CREATE RESP2 values" on page 217 for information on RESP2 values.

+ **200** The command was executed in a program defined with an EXECUTIONSET value of DPLSUBSET or a program invoked from a remote system by a distributed program link without the SYNCONRETURN option.

+ LENGERR

+ RESP2 values:

+ **1** The length you have specified in ATTRLEN is negative.

+ NOTAUTH

+ RESP2 values:

+ **100** The user associated with the issuing task is not authorized to use this command.

+ CREATE MAPSET

APAR PQ02462

added EXEC CICS CREATE to CICS 4.1

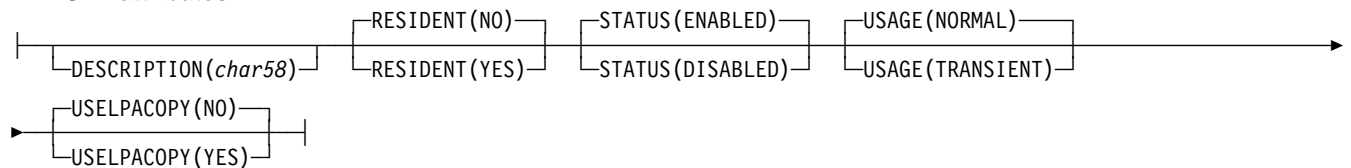
+ Function

+ Define a map set in the local CICS region.

```
▶▶ CREATE MAPSET(data-value)—ATTRIBUTES(data-value)—ATTRLEN(data-value)—▶▶
```

+ **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH

MAPSET attributes:



APAR PQ09156

added this note

Note to COBOL programmers: In the syntax above, you
must use --- ATTRIBUTES (*data-area*)--- instead of
--- ATTRIBUTES (*data-value*)--- .

+ Usage

+ The CREATE MAPSET command builds a MAPSET definition, without reference to data stored on the CSD file.
+ Map set names must be unique among map set, program, and partition set names within a CICS region. If the local region already has one of these resources by the name you specify, the new definition replaces the old one; if not, the new definition is added.

+ A syncpoint is implicit in CREATE MAPSET processing, except when an exception condition is detected early in processing the command. Uncommitted changes to recoverable resources made up to that point in the task are recommitted if the CREATE executes successfully and rolled back if not. See “Creating resource definitions” on page 16 for other general rules governing CREATE commands.

+ Options

+ **ATTRIBUTES**(*data-value*)
+ specifies the attributes of the MAPSET being added.
+ The list of attributes must be coded as a single character string using the syntax shown in **MAPSET attributes**.
+ See “ATTRIBUTES option” on page 16 for general rules

+ for specifying attributes, and the MAPSET chapter in the *CICS/ESA Resource Definition Guide* for details about specific attributes.

+ **Note:** You can assign default values for all attributes of a MAPSET definition by specifying an ATTRLEN value of 0. You still need to specify the ATTRIBUTES option, however, even though its value is not used.

+ **ATTRLEN**(*data-value*)

+ specifies the length in bytes of the character string supplied in the ATTRIBUTES option, as a halfword binary value. The length can be from 0 to 32767.

+ **MAPSET**(*data-value*)

+ specifies the 8-character name of the MAPSET definition to be added to the CICS region.

+ Conditions

+ **ILLOGIC**

+ RESP2 values:

+ **2** The command cannot be executed because an earlier CONNECTION or TERMINAL pool definition has not yet been completed.

+ **INVREQ**

+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES string, or an error occurred during either the discard or resource definition phase of the processing. See Appendix C, “EXEC CICS CREATE RESP2 values” on page 217 for information on RESP2 values.

CREATE MAPSET

+ **200** The command was executed in a program defined
+ with an EXECUTIONSET value of DPLSUBSET or
+ a program invoked from a remote system by a
+ distributed program link without the
+ SYNCONRETURN option.

+ LENGERR

+ RESP2 values:

+ **1** The length you have specified in ATTRLEN is
+ negative.

+ NOTAUTH

+ RESP2 values:

+ **100** The user associated with the issuing task is not
+ authorized to use this command.

+ **101** The user associated with the issuing task is not
+ authorized to create a MAPSET definition with this
+ name.

+ CREATE PARTITIONSET

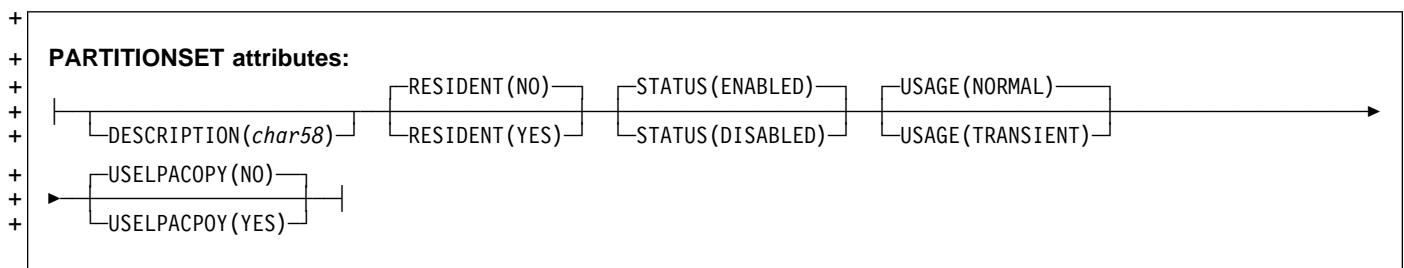
+ **APAR PQ02462**
+ added EXEC CICS CREATE to CICS 4.1

+ Function

+ Define a partition set in the local CICS region.

+ **CREATE PARTITIONSET**(*data-value*)—**ATTRIBUTES**(*data-value*)—**ATTRLEN**(*data-value*)—

+ **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH



APAR PQ09156
added this note

Note to COBOL programmers: In the syntax above, you
must use --- ATTRIBUTES (*data-area*)--- instead of
--- ATTRIBUTES (*data-value*)--- .

+ Usage

+ The CREATE PARTITIONSET command builds a
+ PARTITIONSET definition, without reference to data stored
+ on the CSD file. Partition set names must be unique among
+ partition set, map set and program names within a CICS
+ region. If the local region already has one of these
+ resources by the name you specify, the new definition
+ replaces the old one; if not, the new definition is added.

+ A syncpoint is implicit in CREATE PARTITIONSET
+ processing, except when an exception condition is detected
+ early in processing the command. Uncommitted changes to
+ recoverable resources made up to that point in the task are
+ recommitted if the CREATE executes successfully and rolled
+ back if not. See “Creating resource definitions” on page 16
+ for other general rules governing CREATE commands.

+ Options

+ **ATTRIBUTES**(*data-value*)
+ specifies the attributes of the PARTITIONSET being
+ added. The list of attributes must be coded as a single
+ character string using the syntax shown in
+ **PARTITIONSET attributes**. See “ATTRIBUTES option”

+ on page 16 for general rules for specifying attributes,
+ and the PARTITIONSET chapter in the *CICS/ESA*
+ *Resource Definition Guide* for details about specific
+ attributes.

+ **Note:** You can assign default values for all attributes of
+ a PARTITIONSET definition by specifying an ATTRLEN
+ value of 0. You still need to specify the ATTRIBUTES
+ option, however, even though its value is not used.

+ **ATTRLEN**(*data-value*)

+ specifies the length in bytes of the character string
+ supplied in the ATTRIBUTES option, as a halfword
+ binary value. The length can be from 0 to 32767.

+ **PARTITIONSET**(*data-value*)

+ specifies the 8-character name of the PARTITIONSET
+ definition to be added to the CICS region.

+ Conditions

+ **ILLOGIC**

+ RESP2 values:

+ **2** The command cannot be executed because an
+ earlier CONNECTION or TERMINAL pool
+ definition has not yet been completed.

+ **INVREQ**

+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES
+ string, or an error occurred during either the
+ discard or resource definition phase of the
+ processing. See Appendix C, “EXEC CICS

CREATE PARTITIONSET

+ CREATE RESP2 values" on page 217 for
+ information on RESP2 values.
+ **200** The command was executed in a program defined
+ with an EXECUTIONSET value of DPLSUBSET or
+ a program invoked from a remote system by a
+ distributed program link without the
+ SYNCONRETURN option.

+ LENGERR

+ RESP2 values:

+ **1** The length you have specified in ATTRLEN is
+ negative.

+ NOTAUTH

+ RESP2 values:

+ **100** The user associated with the issuing task is not
+ authorized to use this command.

+ **101** The user associated with the issuing task is not
+ authorized to create a PARTITIONSET definition
+ with this name.

+ CREATE PARTNER**APAR PQ02462**

+ added EXEC CICS CREATE to CICS 4.1

+ Function

+ Define a PARTNER in the local CICS region.

```
▶▶ CREATE PARTNER(data-value)—ATTRIBUTES(data-value)—ATTRLEN(data-value)▶▶
```

+ **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH

PARTNER attributes:

```

┌──────────┐ ┌──────────┐ ┌──────────┐ ┌──────────┐ ┌──────────┐
│ DESCRIPTION(char58) │ │ NETNAME(char8) │ │ NETWORK(char8) │ │ PROFILE(DFHCICSA) │ │ TPNAME(char64) │
└──────────┘ └──────────┘ └──────────┘ └──────────┘ └──────────┘
┌──────────┐ ┌──────────┐
│ PROFILE(char8) │ │ XTPNAME(hex128) │
└──────────┘ └──────────┘

```

APAR PQ09156

added this note

Note to COBOL programmers: In the syntax above, you
must use --- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)--- .

+ ATTRLEN(*data-value*)

+ specifies the length in bytes of the character string
+ supplied in the ATTRIBUTES option, as a halfword
+ binary value. The length may not exceed 32767 bytes.

+ PARTNER(*data-value*)

+ specifies the 8-character name of the PARTNER
+ definition to be added to the CICS region.

+ Usage

+ The CREATE PARTNER command builds a PARTNER
+ definition, without reference to data stored on the CSD file. If
+ there is already a partner by the name you specify in the
+ local CICS region, the new definition replaces the old one; if
+ not, the new definition is added.

+ A syncpoint is implicit in CREATE PARTNER processing,
+ except when an exception condition is detected early in
+ processing the command. Uncommitted changes to
+ recoverable resources made up to that point in the task are
+ committed if the CREATE executes successfully and rolled
+ back if not. See “Creating resource definitions” on page 16
+ for other general rules governing CREATE commands.

+ Options**+ ATTRIBUTES(*data-value*)**

+ specifies the attributes of the PARTNER being added.
+ The list of attributes must be coded as a single character
+ string using the syntax shown in **PARTNER attributes**.
+ See “ATTRIBUTES option” on page 16 for general rules
+ for specifying attributes, and the PARTNER chapter in
+ the *CICS/ESA Resource Definition Guide* for details
+ about specific attributes.

+ Conditions**+ ILLOGIC**

+ RESP2 values:

+ **2** The command cannot be executed because an
+ earlier CONNECTION or TERMINAL pool
+ definition has not yet been completed.

+ INVREQ

+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES
+ string, or an error occurred during either the
+ discard or resource definition phase of the
+ processing. See Appendix C, “EXEC CICS
+ CREATE RESP2 values” on page 217 for
+ information on RESP2 values.

+ **200** The command was executed in a program defined
+ with an EXECUTIONSET value of DPLSUBSET or
+ a program invoked from a remote system by a
+ distributed program link without the
+ SYNCONRETURN option.

+ LENGERR

+ RESP2 values:

+ **1** The length you have specified in ATTRLEN is
+ negative.

CREATE PARTNER

+ **NOTAUTH**

+ RESP2 values:

+ **100** The user associated with the issuing task is not authorized to use this command.

+ **CREATE PROFILE**

+ **APAR PQ02462**

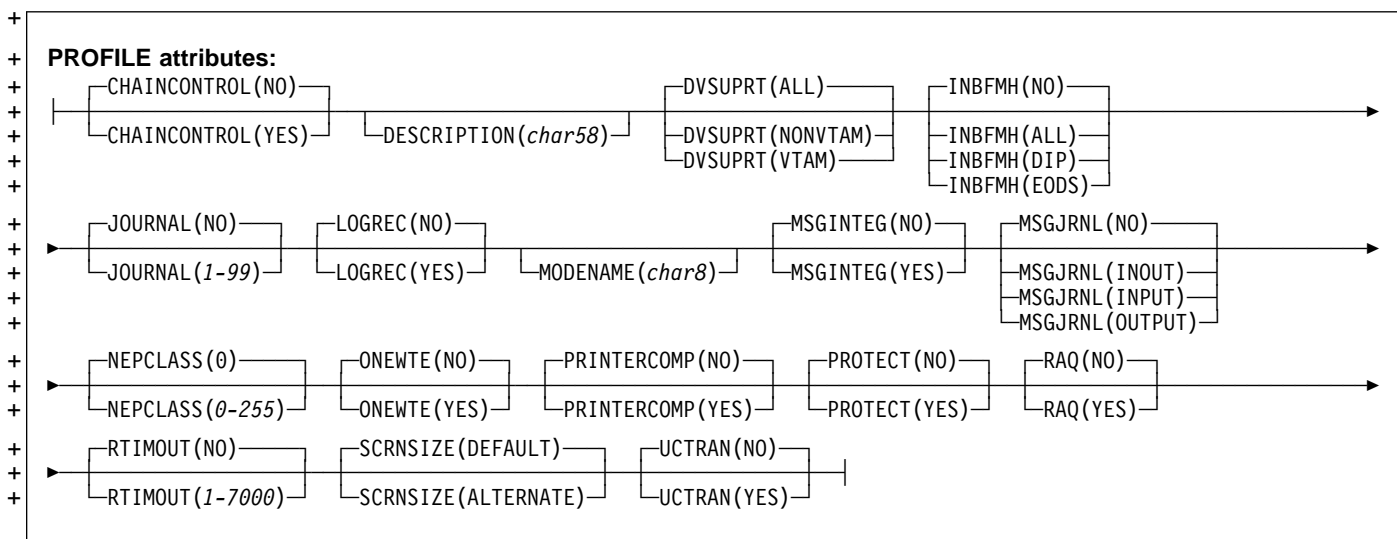
+ added EXEC CICS CREATE to CICS 4.1

+ **Function**

+ Define a transaction PROFILE in the local CICS region.

+ **CREATE PROFILE**(data-value)---**ATTRIBUTES**(data-value)---**ATTRLEN**(data-value)---

+ **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH



APAR PQ09156

added this note

Note to COBOL programmers: In the syntax above, you must use --- ATTRIBUTES (data-area)--- instead of --- ATTRIBUTES (data-value)--- .

+ **Usage**

+ The CREATE PROFILE command builds a PROFILE definition, without reference to data stored on the CSD file. If there is already a profile by the name you specify in the local CICS region, the new definition replaces the old one; if not, the new definition is added.

+ A syncpoint is implicit in CREATE PROFILE processing, except when an exception condition is detected early in processing the command. Uncommitted changes to recoverable resources made up to that point in the task are committed if the CREATE executes successfully and rolled back if not. See "Creating resource definitions" on page 16 for other general rules governing CREATE commands.

+ **Options**

+ **ATTRIBUTES**(data-value)
 + specifies the attributes of the PROFILE being added.
 + The list of attributes must be coded as a single character string using the syntax shown in **PROFILE attributes**.
 + See "ATTRIBUTES option" on page 16 for general rules for specifying attributes, and the PROFILE chapter in the *CICS/ESA Resource Definition Guide* for details about specific attributes.

+ **Note:** You can assign default values for all attributes of a PROFILE definition by specifying an ATTRLEN value of 0. You still need to specify the ATTRIBUTES option, however, even though its value is not used.

+ **ATTRLEN**(data-value)
 + specifies the length in bytes of the character string supplied in the ATTRIBUTES option, as a halfword binary value. The length can be from 0 to 32767.

+ **PROFILE**(data-value)
 + specifies the 8-character name of the PROFILE definition to be added to the CICS region.

CREATE PROFILE

+ Conditions

+ ILLOGIC

+ RESP2 values:

- + **2** The command cannot be executed because an earlier CONNECTION or TERMINAL pool definition has not yet been completed.

+ INVREQ

+ RESP2 values:

- + **n** There is a syntactical error in the ATTRIBUTES string, or an error occurred during either the discard or resource definition phase of the processing. See Appendix C, "EXEC CICS CREATE RESP2 values" on page 217 for information on RESP2 values.
- + **200** The command was executed in a program defined with an EXECUTIONSET value of DPLSUBSET or a program invoked from a remote system by a distributed program link without the SYNCONRETURN option.

+ LENGERR

+ RESP2 values:

- + **1** The length you have specified in ATTRLEN is negative.

+ NOTAUTH

+ RESP2 values:

- + **100** The user associated with the issuing task is not authorized to use this command.

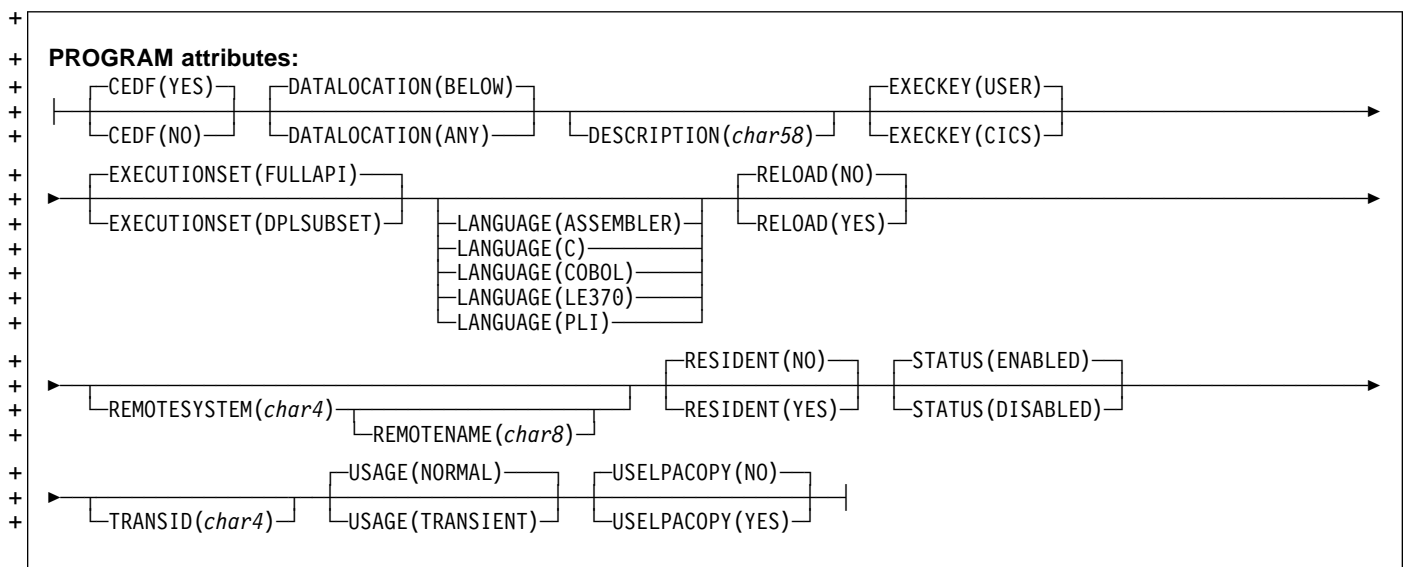
+ **CREATE PROGRAM**

+ **APAR PQ02462**
 + added EXEC CICS CREATE to CICS 4.1

+ **Function**

+ Define a PROGRAM in the local CICS region.

+ **CREATE PROGRAM**(data-value)---**ATTRIBUTES**(data-value)---**ATTRLEN**(data-value)---
 + **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH



APAR PQ09156
 # added this note

+ committed if the CREATE executes successfully and rolled back if not. See "Creating resource definitions" on page 16 for other general rules governing CREATE commands.

Note to COBOL programmers: In the syntax above, you must use --- ATTRIBUTES (data-area)--- instead of --- ATTRIBUTES (data-value)--- .

+ **Usage**

+ The CREATE PROGRAM command builds a PROGRAM definition, without reference to data stored on the CSD file. Program names must be unique among program, map set, and partition set names within a CICS region. If the local region already has one of these resources by the name you specify, the new definition replaces the old one; if not, the new definition is added.

+ A syncpoint is implicit in CREATE PROGRAM processing, except when an exception condition is detected early in processing the command. Uncommitted changes to recoverable resources made up to that point in the task are

+ **Options**

+ **ATTRIBUTES**(data-value)

+ specifies the attributes of the PROGRAM being added. The list of attributes must be coded as a single character string using the syntax shown in **PROGRAM attributes**. See "ATTRIBUTES option" on page 16 for general rules for specifying attributes, and the PROGRAM chapter in the *CICS/ESA Resource Definition Guide* for details about specific attributes.

+ **Note:** You can assign default values for all attributes of a PROGRAM definition by specifying an ATTRLEN value of 0. You still need to specify the ATTRIBUTES option, however, even though its value is not used.

+ **ATTRLEN**(data-value)

+ specifies the length in bytes of the character string supplied in the ATTRIBUTES option, as a halfword binary value. The length can be from 0 to 32767.

CREATE PROGRAM

- + **PROGRAM**(*data-value*)
- + specifies the 8-character name of the PROGRAM
- + definition to be added to the CICS region.

+ Conditions

+ ILLOGIC

- + RESP2 values:
- + **2** The command cannot be executed because an
- + earlier CONNECTION or TERMINAL pool
- + definition has not yet been completed.

+ INVREQ

- + RESP2 values:
- + **n** There is a syntactical error in the ATTRIBUTES
- + string, or an error occurred during either the
- + discard or resource definition phase of the
- + processing. See Appendix C, "EXEC CICS

- + CREATE RESP2 values" on page 217 for
- + information on RESP2 values.
- + **200** The command was executed in a program defined
- + with an EXECUTIONSET value of DPLSUBSET or
- + a program invoked from a remote system by a
- + distributed program link without the
- + SYNCONRETURN option.

+ LENGERR

- + RESP2 values:
- + **1** The length you have specified in ATTRLEN is
- + negative.

+ NOTAUTH

- + RESP2 values:
- + **100** The user associated with the issuing task is not
- + authorized to use this command.
- + **101** The user associated with the issuing task is not
- + authorized to create a PROGRAM definition with
- + this name.

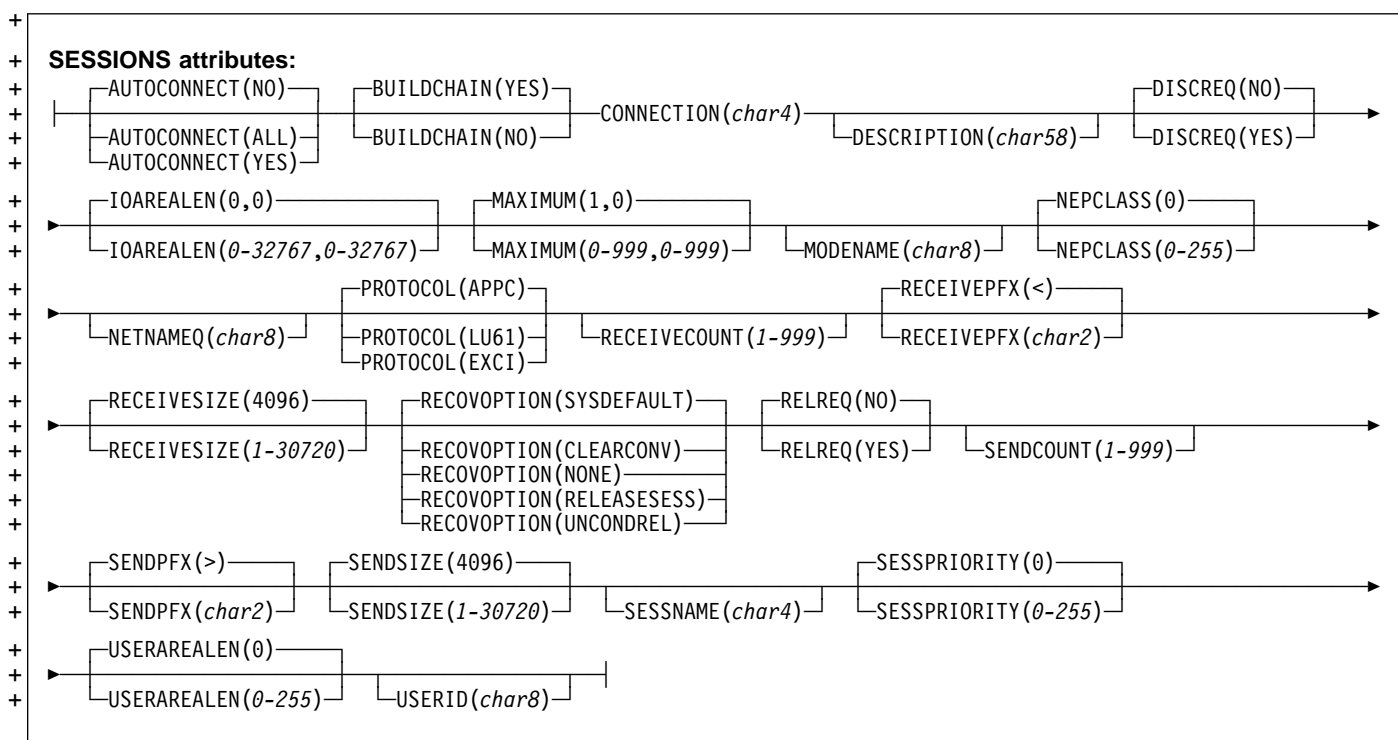
+ **CREATE SESSIONS**

+ **APAR PQ02462**
 + added EXEC CICS CREATE to CICS 4.1

+ **Function**

+ Add a session group to the CONNECTION definition being created.

+ **CREATE SESSIONS**(*data-value*)—**ATTRIBUTES**(*data-value*)—**ATTRLEN**(*data-value*)—**◆◆**
 + **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH



APAR PQ09156
 # added this note

Note to COBOL programmers: In the syntax above, you
 # must use --- ATTRIBUTES (data-area)--- instead of
 # --- ATTRIBUTES (data-value)--- .

+ **Usage**

+ The CREATE SESSIONS command defines a group of sessions within a CONNECTION definition under construction, without reference to data stored on the CSD file. You can use it only after issuing the initial CREATE CONNECTION command that defines the attributes of a connection and before the final CREATE CONNECTION COMPLETE (or DISCARD) command that ends the process.

+ The sessions you define always belong to the current connection, and the name that you specify in the CONNECTION option within your ATTRIBUTES string must match the name of the connection specified in the preceding CREATE CONNECTION command. See the discussion of the CREATE CONNECTION command on page 27 for rules about the order of the commands that build a connection, and “Creating resource definitions” on page 16 for general rules governing CREATE commands.

+ **Options**

+ **ATTRIBUTES**(*data-value*)
 + specifies the attributes of the group of sessions being added. The list of attributes must be coded as a single character string using the syntax shown in **SESSIONS attributes**. See “ATTRIBUTES option” on page 16 for general rules for specifying attributes, and the

CREATE SESSIONS

- + SESSIONS chapter in the *CICS/ESA Resource Definition Guide* for details about specific attributes.

+ ATTRLEN(*data-value*)

- + specifies the length in bytes of the character string supplied in the ATTRIBUTES option, as a halfword binary value. The length may not exceed 32767 bytes.

+ SESSIONS(*data-value*)

- + specifies the 8-character name of the SESSIONS definition to be added to CONNECTION definition under construction. The name of a sessions group needs to be unique only within the current CONNECTION definition, and the group is always added unless you repeat a session name within a connection. In this case, the last successful SESSIONS definition of the same name is the one that is used.

+ Conditions

+ ILLOGIC

- + RESP2 values:

- + **2** The command cannot be executed because no CREATE CONNECTION ATTRIBUTES command has been issued, or the CONNECTION name specified in the ATTRIBUTES argument of this command does not match the name of the connection assigned in the CREATE CONNECTION command.

+ INVREQ

- + RESP2 values:

- + **n** There is a syntactical error in the ATTRIBUTES string, or an error occurred during either the discard or resource definition phase of the processing. See Appendix C, "EXEC CICS CREATE RESP2 values" on page 217 for information on RESP2 values.
- + **200** The command was executed in a program defined with an EXECUTIONSET value of DPLSUBSET or a program invoked from a remote system by a distributed program link without the SYNCONRETURN option.

+ LENGERR

- + RESP2 values:

- + **1** The length you have specified in ATTRLEN is negative.

+ NOTAUTH

- + RESP2 values:

- + **100** The user associated with the issuing task is not authorized to use this command.
- + **102** The user of the transaction issuing the CREATE SESSIONS command is not an authorized surrogate of the user specified in USERID.

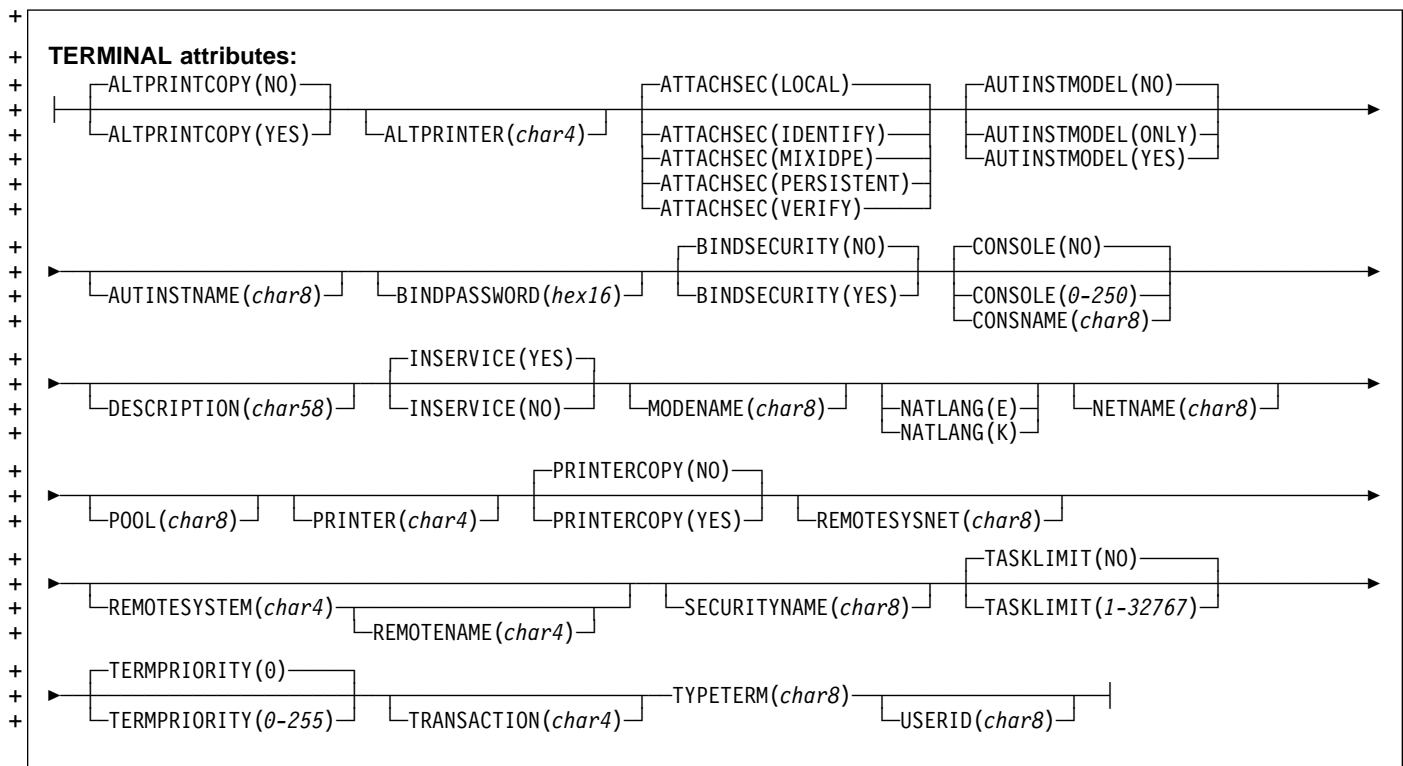
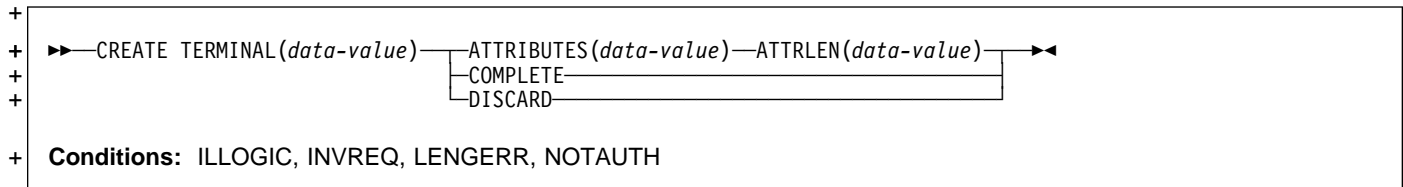
+ **CREATE TERMINAL**

+ **APAR PQ02462**

+ added EXEC CICS CREATE to CICS 4.1

+ **Function**

+ Define a TERMINAL in the local CICS region.



APAR PQ09156

added this note

Note to COBOL programmers: In the syntax above, you
 # must use --- ATTRIBUTES (data-area)--- instead of
 # --- ATTRIBUTES (data-value)--- .

+ **Usage**

+ CREATE TERMINAL commands build TERMINAL definitions,
 + without reference to data stored on the CSD file. You can
 + use them either to define individual terminals or a pool of
 + terminals.

+ The POOL attribute determines which mode you are using.
 + Without it, each command defines a single, independent
 + terminal. If there is already a terminal by the name you
 + specify in the local CICS region, the new definition replaces
 + the old one; if not, the new definition is added.

+ To define a pool, you issue one CREATE TERMINAL
 + ATTRIBUTES command for each terminal in the pool,
 + specifying the same POOL value in the ATTRIBUTES string.

CREATE TERMINAL

+ After all of the terminals are defined, you issue CREATE
+ TERMINAL COMPLETE; CICS collects but does not install
+ the TERMINAL definitions until the COMPLETE command.
+ At this point, if there was a pool of the same name in the
+ local CICS region, CICS deletes all of its terminals and
+ installs the new definitions; if not, it adds the new definitions.
+ Consequently, pool terminals must be defined all at once;
+ you cannot add terminals to an existing pool or include a
+ terminal with the same name as an existing non-pool
+ terminal.

+ During the time the pool is being built, you may not:

- + • Change or omit the pool name
- + • Define other resources of any type, including terminals
+ outside the current pool
- + • Issue a SYNCPOINT (or any command that implies one)
- + • Terminate your task (normally)

+ However, if you encounter an error or problem during the
+ course of building a pool, you can terminate the process at
+ any point by issuing a CREATE TERMINAL DISCARD
+ command. If you do this, CICS will discard the partial pool
+ definition, including all of its terminals.

+ A syncpoint is implicit in CREATE TERMINAL processing, as
+ in other CREATE commands, except when an exception
+ condition is detected early in the processing. Uncommitted
+ changes to recoverable resources are committed when
+ definitions are processed successfully and rolled back if not
+ or if you specify DISCARD. For non-pool terminals, the
+ syncpoint occurs on each CREATE command. When you
+ are building a pool, however, it occurs only on the command
+ that ends the pool definition, whether you specify
+ COMPLETE or DISCARD. See “Creating resource
+ definitions” on page 16 for other general rules governing
+ CREATE commands.

+ Options

+ **ATTRIBUTES**(*data-value*)
+ specifies the attributes of the TERMINAL being added.
+ The list of attributes must be coded as a single character
+ string using the syntax shown in **TERMINAL attributes**.
+ See “ATTRIBUTES option” on page 16 for general rules
+ for specifying attributes, and the TERMINAL chapter in
+ the *CICS/ESA Resource Definition Guide* for details
+ about specific attributes.

+ **ATTRLEN**(*data-value*)
+ specifies the length in bytes of the character string
+ supplied in the ATTRIBUTES option, as a halfword
+ binary value. The length may not exceed 32767 bytes.

+ **COMPLETE**
+ specifies that the terminal pool definition under
+ construction is complete. It can be used only after the
+ last terminal of a pool has been defined.

+ **TERMINAL**(*data-value*)
+ specifies the 4-character name of the TERMINAL
+ definition to be added.

+ **DISCARD**
+ specifies that the terminal pool definition under
+ construction will not be completed and all of the
+ TERMINAL definitions issued since the pool was started
+ are to be discarded and *not* added.

+ Conditions

+ **ILLOGIC**
+ RESP2 values:

+ **2** The command cannot be executed because an
+ earlier CONNECTION or TERMINAL pool
+ definition has not yet been completed.

+ **INVREQ**
+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES
+ string, or an error occurred during either the
+ discard or resource definition phase of the
+ processing. See Appendix C, “EXEC CICS
+ CREATE RESP2 values” on page 217 for
+ information on RESP2 values.
+ **200** The command was executed in a program defined
+ with an EXECUTIONSET value of DPLSUBSET or
+ a program invoked from a remote system by a
+ distributed program link without the
+ SYNCONRETURN option.

+ **LENGERR**
+ RESP2 values:

+ **1** The length specified in ATTRLEN is negative.

+ **NOTAUTH**
+ RESP2 values:

+ **100** The user associated with the issuing task is not
+ authorized to use this command.
+ **102** The user associated with the task issuing the
+ CREATE TERMINAL command is not an
+ authorized surrogate of the user specified in
+ USERID.

+ CREATE TRANCLASS

APAR PQ02462

added EXEC CICS CREATE to CICS 4.1

+ Function

+ Define a transaction class in the local CICS region.

```
▶▶ CREATE TRANCLASS(data-value)—ATTRIBUTES(data-value)—ATTRLEN(data-value)▶▶
```

Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

TRANCLASS attributes:

```
┌──────────┴───┐ ┌──────────┴───┐
DESCRIPTION(char58) MAXACTIVE(0-999) ┌───┴───┐
└──────────┬───┘ └───┬───┘ ┌───┴───┐
└───┬───┘ └───┬───┘ └───┬───┘
PURGETHRESH(NO)
PURGETHRESH(1-1000000)
```

APAR PQ09156

added this note

Note to COBOL programmers: In the syntax above, you
must use --- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)--- .

+ **ATTRLEN**(*data-value*)

+ specifies the length in bytes of the character string
+ supplied in the ATTRIBUTES option, as a halfword
+ binary value. The length may not exceed 32767 bytes.

+ **TRANCLASS**(*data-value*)

+ specifies the 8-character name of the TRANCLASS
+ definition to be added to the CICS region.

+ Usage

+ The CREATE TRANCLASS command builds a TRANCLASS
+ definition, without reference to data stored on the CSD file. If
+ there is already a transaction class by the name you specify
+ in the local CICS region, the new definition replaces the old
+ one; if not, the new definition is added.

+ A syncpoint is implicit in CREATE TRANCLASS processing,
+ except when an exception condition is detected early in
+ processing the command. Uncommitted changes to
+ recoverable resources made up to that point in the task are
+ committed if the CREATE executes successfully and rolled
+ back if not. See "Creating resource definitions" on page 16
+ for other general rules governing CREATE commands.

+ Options

+ **ATTRIBUTES**(*data-value*)

+ specifies the attributes of the TRANCLASS being added.
+ The list of attributes must be coded as a single character
+ string using the syntax shown in **TRANCLASS**
+ **attributes**. See "ATTRIBUTES option" on page 16 for
+ general rules for specifying attributes, and the
+ TRANCLASS chapter in the *CICS/ESA Resource*
+ *Definition Guide* for details about specific attributes.

+ Conditions

+ **ILLOGIC**

+ RESP2 values:

+ **2** The command cannot be executed because an
+ earlier CONNECTION or TERMINAL pool
+ definition has not yet been completed.

+ **INVREQ**

+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES
+ string, or an error occurred during either the
+ discard or resource definition phase of the
+ processing. See Appendix C, "EXEC CICS
+ CREATE RESP2 values" on page 217 for
+ information on RESP2 values.

+ **200** The command was executed in a program defined
+ with an EXECUTIONSET value of DPLSUBSET or
+ a program invoked from a remote system by a
+ distributed program link without the
+ SYNCONRETURN option.

+ **LENGERR**

+ RESP2 values:

+ **1** The length you have specified in ATTRLEN is
+ negative.

CREATE TRANCLASS

+ **NOTAUTH**

+ RESP2 values:

+ **100** The user associated with the issuing task is not
+ authorized to use this command.

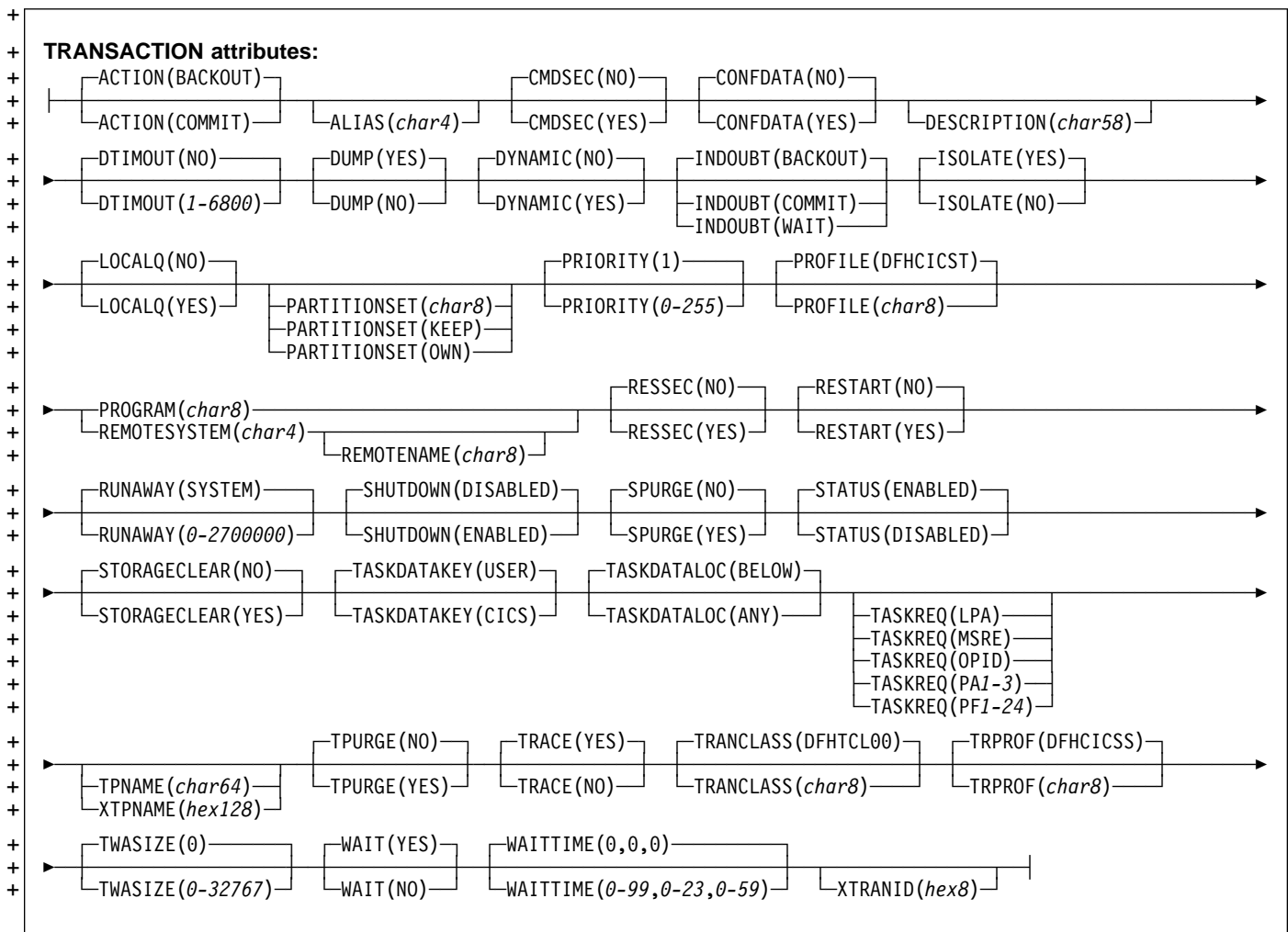
+ **CREATE TRANSACTION**

+ **APAR PQ02462**
 + added EXEC CICS CREATE to CICS 4.1

+ **Function**

+ Define a TRANSACTION in the local CICS region.

+ **CREATE TRANSACTION**(*data-value*)—**ATTRIBUTES**(*data-value*)—**ATTRLEN**(*data-value*)—**▶**
 + **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH



APAR PQ09156
 # added this note

Note to COBOL programmers: In the syntax above, you must use --- ATTRIBUTES (data-area)--- instead of --- ATTRIBUTES (data-value)--- .

+ **Usage**

+ The CREATE TRANSACTION command builds a TRANSACTION definition, without reference to data stored on the CSD file. If there is no transaction by the name you specify in the local CICS region, the new definition is added. If there is, the new definition replaces the old one. However, it does not apply to tasks already in flight, which continue to use the definition under which they were initiated.

CREATE TRANSACTION

+ A syncpoint is implicit in CREATE TRANSACTION
+ processing, except when an exception condition is detected
+ early in processing the command. Uncommitted changes to
+ recoverable resources made up to that point in the task are
+ committed if the CREATE executes successfully and rolled
+ back if not. See “Creating resource definitions” on page 16
+ for other general rules governing CREATE commands.

+ Options

+ **ATTRIBUTES**(*data-value*)

+ specifies the attributes of the TRANSACTION being
+ added. The list of attributes must be coded as a single
+ character string using the syntax shown in
+ **TRANSACTION attributes**. See “ATTRIBUTES option”
+ on page 16 for general rules for specifying attributes,
+ and the TRANSACTION chapter in the *CICS/ESA*
+ *Resource Definition Guide* for details about specific
+ attributes.

+ **ATTRLEN**(*data-value*)

+ specifies the length in bytes of the character string
+ supplied in the ATTRIBUTES option, as a halfword
+ binary value. The length may not exceed 32767 bytes.

+ **TRANSACTION**(*data-value*)

+ specifies the 4-character name of the TRANSACTION
+ definition to be added to the CICS region.

+ Conditions

+ **ILLOGIC**

+ RESP2 values:

+ **2** The command cannot be executed because an
+ earlier CONNECTION or TERMINAL pool
+ definition has not yet been completed.

+ **INVREQ**

+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES
+ string, or an error occurred during either the
+ discard or resource definition phase of the
+ processing. See Appendix C, “EXEC CICS
+ CREATE RESP2 values” on page 217 for
+ information on RESP2 values.

+ **200** The command was executed in a program defined
+ with an EXECUTIONSET value of DPLSUBSET or
+ a program invoked from a remote system by a
+ distributed program link without the
+ SYNCONRETURN option.

+ **LENGERR**

+ RESP2 values:

+ **1** The length you have specified in ATTRLEN is
+ negative.

+ **NOTAUTH**

+ RESP2 values:

+ **100** The user associated with the issuing task is not
+ authorized to use this command.
+ **101** The user associated with the issuing task is not
+ authorized to create a TRANSACTION definition
+ with this name.

+ CREATE TYPETERM

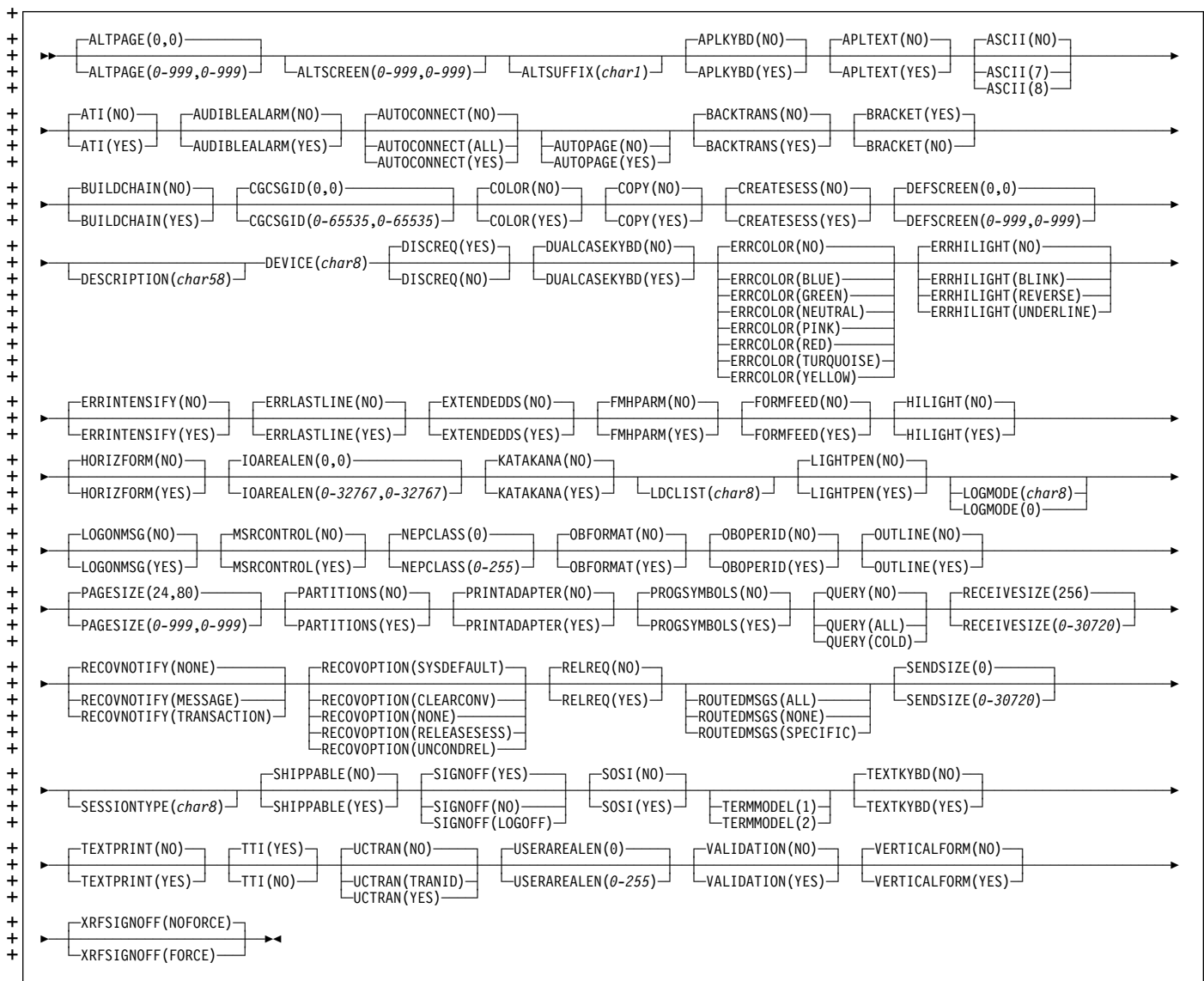
APAR PQ02462
 + added EXEC CICS CREATE to CICS 4.1

+ Function

+ Define a terminal type in the local CICS region.

+ **►►** CREATE TYPETERM(*data-value*)—ATTRIBUTES(*data-value*)—ATTRLEN(*data-value*)—**►►**

+ **Conditions:** ILLOGIC, INVREQ, LENGERR, NOTAUTH



APAR PQ09156
 # added this note

Note to COBOL programmers: In the syntax above, you must use --- ATTRIBUTES (*data-area*)--- instead of --- ATTRIBUTES (*data-value*)--- .

CREATE TYPETERM

+ Usage

+ The CREATE TYPETERM command builds a terminal type (TYPETERM) definition, without reference to data stored on the CSD file. If there is already a terminal type definition by the name you specify in the local CICS region, the new definition replaces the old one; if not, the new definition is added.

+ A syncpoint is implicit in CREATE TYPETERM processing, except when an exception condition is detected early in processing the command. Uncommitted changes to recoverable resources made up to that point in the task are committed if the CREATE executes successfully and rolled back if not. See “Creating resource definitions” on page 16 for other general rules governing CREATE commands.

+ Options

+ ATTRIBUTES(*data-value*)

+ specifies the attributes of the TYPETERM being added. The list of attributes must be coded as a single character string using the syntax shown in **TYPETERM attributes**. See “ATTRIBUTES option” on page 16 for general rules for specifying attributes, and the TYPETERM chapter in the *CICS/ESA Resource Definition Guide* for details about specific attributes.

+ ATTRLEN(*data-value*)

+ specifies the length in bytes of the character string supplied in the ATTRIBUTES option, as a halfword binary value. The length may not exceed 32767 bytes.

+ TYPETERM(*data-value*)

+ specifies the 8-character name of the TYPETERM definition to be added to the CICS region.

+ Conditions

+ ILLOGIC

+ RESP2 values:

+ **2** The command cannot be executed because an earlier CONNECTION or TERMINAL pool definition has not yet been completed.

+ INVREQ

+ RESP2 values:

+ **n** There is a syntactical error in the ATTRIBUTES string, or an error occurred during either the discard or resource definition phase of the processing. See Appendix C, “EXEC CICS CREATE RESP2 values” on page 217 for information on RESP2 values.

+ **200** The command was executed in a program defined with an EXECUTIONSET value of DPLSUBSET or a program invoked from a remote system by a distributed program link without the SYNCONRETURN option.

+ LENGERR

+ RESP2 values:

+ **1** The length you have specified in ATTRLEN is negative.

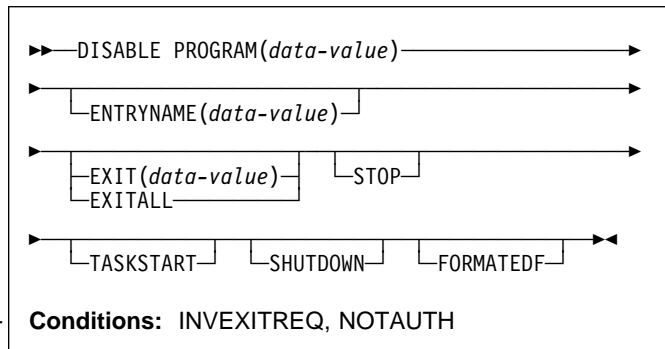
+ NOTAUTH

+ RESP2 values:

+ **100** The user associated with the issuing task is not authorized to use this command.

DISABLE PROGRAM

Terminate or otherwise modify the invocation of a user exit.



Description

The DISABLE PROGRAM command changes the status of a global or task-related user exit, reversing the effects of corresponding options in an ENABLE PROGRAM command.

You use it to:

- Remove points at which a particular exit is invoked
- Make the exit unavailable for execution (without removing its status as an exit)
- Delete its definition as an exit entirely.

Options on the DISABLE PROGRAM command correspond to those on the ENABLE command:

- ENTRYNAME and PROGRAM identify the exit to be disabled, and you must use exactly the same combination of values that you did in the ENABLE command that defined the exit.
- EXIT, FORMATEDF, SHUTDOWN, and TASKSTART reverse the effect of the same-named options on ENABLE PROGRAM; that is, they turn off invocation of the exit at the points specified.
- STOP reverses the effect of START, making the exit unavailable for execution.
- EXITALL deletes the definition entirely, reversing the effect of the ENABLE PROGRAM that created the exit. Work areas and the load module associated with the exit may be deleted as well.

For programming information about CICS exits, see the *CICS/ESA Customization Guide*; you should also see the general discussion of commands that modify exits in “Exit-related commands” on page 18.

Note: One or more of STOP and (EXIT or EXITALL) is required for a global user exit, and one or more of STOP, EXITALL, TASKSTART, SHUTDOWN, and FORMATEDF is required for a task-related user exit.

Options

ENTRYNAME(data-value)

specifies the name of the global or task-related user exit whose status is to be changed. If you omit ENTRYNAME, CICS assumes that the name of the exit is the same as the load module name given in the PROGRAM option. Therefore, you must use the same combination of ENTRYNAME and PROGRAM values on DISABLE commands as was specified on the initial ENABLE command that defined the exit.

EXIT(data-value) (global user exits only)

specifies the name of the global user exit point from which this exit is to be dissociated. It causes CICS to stop invoking the exit at this point but does not, of itself, cause CICS to delete the associated load module from virtual storage, even if it is no longer being used at any exit points. Exit point names are 8 characters long; for programming information, including a list of exit points, see the *CICS/ESA Customization Guide*.

EXITALL

causes CICS to discard the definition of the exit. For a global user exit, EXITALL dissociates the exit from all of the exit points from which it currently is invoked. If possible, the associated load module is deleted from virtual storage.

For a task-related user exit, the associated load module is deleted from virtual storage if it is not in use by another exit and if the ENTRY option was not specified in the ENABLE command that defined the exit. If the exit owns a global work area, the work area is released as soon as no other exits are sharing it.

EXITALL implies STOP, so the exit becomes unavailable for execution. For a task-related user exit, you must avoid requesting this function until all tasks that have used the exit have ended; the results of EXITALL before that point are unpredictable.

FORMATEDF (task-related user exits only)

indicates that the exit should not be invoked to format EDF screens. You can reinstate invocation at EDF points with an ENABLE command specifying FORMATEDF.

PROGRAM(data-value)

specifies the 8-character name of the load module that contains the entry point for the exit. This name is also used as the name of the exit when ENTRYNAME is not specified; see the ENTRYNAME option.

SHUTDOWN (task-related user exits only)

indicates that the exit should not be invoked at CICS shutdown. You can reinstate invocation at shutdown with an ENABLE command specifying SHUTDOWN.

DISABLE PROGRAM

STOP

specifies that the exit is to be made unavailable for execution, but is to remain enabled (defined as an exit). You can make the exit available for execution again with an ENABLE command specifying START.

When a STOPped task-related user exit gets invoked, the invoking code gets an AEY9 abend code. There is no corresponding error for global user exits, however, because CICS invokes only those exits associated with an exit point which are also available for execution (not stopped).

TASKSTART (task-related user exits only)

indicates that the exit should not be invoked at the start and end of each task. You can reinstate these invocations with an ENABLE command specifying TASKSTART.

Conditions

+ INVEXITREQ

- + The INVEXITREQ condition of the DISABLE command is indicated by X'80' in the first byte of EIBRCODE.
- + The exact cause of the error can be determined by examining the second and third bytes of EIBRCODE, which can have the values shown in the following list.
- + **X'808000'** The load module named on the PROGRAM parameter has not been defined to CICS, or the load module is not in the load library, or the load module has been disabled.
- + **X'804000'** The value of EXIT is not a valid exit point.
- + **X'800200'** The exit identified by the PROGRAM value is not defined as an exit.
- + **X'800100'** The exit identified by ENTRYNAME is not defined as an exit.
- + **X'800080'** The exit is currently invoked by another task (see note).

+ **Note:** The INVEXITREQ condition with X'0080' in the second and third bytes can occur:

- + • If you issue the DISABLE request while a task using the exit has been suspended temporarily because of a request for a CICS service within the exit. The normal action for this condition is to retry the DISABLE request.
- + • When a DISABLE request with EXITALL or EXIT has been specified, but the exit has already terminated abnormally. In this case, the use count of the associated load module remains greater than zero. The exit cannot be dissociated from any exit point, and the load module cannot be deleted from virtual storage. The exit can, however, be made unavailable for execution by issuing a DISABLE STOP command.

+ NOTAUTH

+ RESP2 values:

- + **100** The user associated with the issuing task is not authorized to use this command.
- + **101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

Examples

Example 1

```
EXEC CICS DISABLE PROGRAM('EP2') STOP
```

Example 1 makes exit EP2 non-executable. It does not dissociate it from the exit points with which it is associated, however, or delete its definition as an exit. It can be made available again by issuing an ENABLE PROGRAM('EP2') START command.

Example 2

```
EXEC CICS DISABLE ENTRYNAME ('ZX') PROGRAM('EP3')  
EXIT('XTDREQ')
```

Example 2 stops global user exit ZX from being invoked at exit point XTDREQ. ZX is still defined, however, and if it is associated with other exit points, it will still be invoked at them.

Example 3

```
EXEC CICS DISABLE PROGRAM('EP3') EXITALL
```

Example 3 dissociates EP3 from all points at which invocation was requested (exit points, in the case of a global user exit; task start, shutdown, and so on, in the case of a task-related user exit), and discards the definition of the exit. If the load module EP3 is not in use, it will be deleted.

DISCARD AUTINSTMODEL

Remove a terminal autoinstall model definition.

```
▶▶—DISCARD AUTINSTMODEL(data-value)—◀◀
```

Conditions: INVREQ, MODELIDERR, NOTAUTH

Description

The DISCARD AUTINSTMODEL command removes the definition of an autoinstall model from the executing CICS system, so that the system cannot use it as a model for automatic installation of terminals.

See “Discarding resource definitions” on page 17 for general information about discards.

Options

AUTINSTMODEL(*data-value*)

specifies the 8-character name of the autoinstall model that is to be removed. This is the name specified in the AUTINSTNAME option of the TERMINAL definition that defines the model. Note that the TERMINAL definition may be installed as a terminal as well as a model. In this case, only the model is discarded.

Models whose names begin with the letters DFH are assumed to be CICS-supplied models and cannot be discarded.

Conditions

INVREQ

RESP2 values:

- 2 The model you requested is currently in use.
- 3 The model cannot be discarded because its name begins with DFH.

MODELIDERR

RESP2 values:

- 1 The model cannot be found.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

DISCARD FILE

Remove a file definition.

```
▶▶—DISCARD FILE(data-value)—◀◀
```

Conditions: FILENOTFOUND, INVREQ, NOTAUTH

Description

The DISCARD FILE command removes the definition of a file from the executing CICS system, so that the system no longer has access to the file. That is, it revokes the earlier installation of a FILE resource definition of the same name. A file must be closed, disabled, and not currently in use for its definition to be discarded.

See “Discarding resource definitions” on page 17 for general information about discards.

Options

FILE(*data-value*)

specifies the 8-character name of the file that is to be removed. You cannot remove the definition of a file whose name begins with the letters DFH, because such files are reserved for CICS.

Conditions

FILENOTFOUND

RESP2 values:

- 18 The file cannot be found.

INVREQ

RESP2 values:

- 2 The file is not closed.
- 3 The file is not disabled.
- 25 The file definition is currently in use.
- 26 The file cannot be discarded because its name begins with DFH.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.
- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

DISCARD PARTNER

DISCARD PARTNER

Remove a partner definition.

```
▶▶—DISCARD PARTNER(data-value)—◀◀
```

Conditions: INVREQ, NOTAUTH, PARTNERIDERR

Description

The DISCARD PARTNER command removes the definition of a partner from the executing CICS system, so that the system no longer has access to the partner. That is, it revokes the earlier installation of a PARTNER resource definition of the same name.

See “Discarding resource definitions” on page 17 for general information about discards.

Options

PARTNER(*data-value*)

specifies the 8-character name of the partner that is to be removed. It must not begin with the letters DFH, because such partners are CICS-defined and may not be deleted.

Conditions

INVREQ

RESP2 values:

- 2 The partner definition is currently in use.
- 3 The partner cannot be discarded because its name begins with DFH.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

PARTNERIDERR

RESP2 values:

- 1 The partner cannot be found.
- 5 The Partner Resource Manager (PRM) is not active, because it failed to initialize during CICS initialization.

DISCARD PROFILE

Remove a profile definition.

```
▶▶—DISCARD PROFILE(data-value)—◀◀
```

Conditions: INVREQ, NOTAUTH, PROFILEIDERR

Description

The DISCARD PROFILE command removes the definition of a profile from the executing CICS system, so that the system no longer has access to the profile. That is, it revokes the earlier installation of a PROFILE resource definition of the same name. You cannot discard a profile while any installed TRANSACTION definitions point to it.

See “Discarding resource definitions” on page 17 for general information about discards.

Options

PROFILE(*data-value*)

specifies the 8-character name of the profile that is to be removed. It must not begin with the letters DFH, because such profiles are CICS-defined and cannot be discarded.

Conditions

INVREQ

RESP2 values:

- 2 The profile definition is currently in use.
- 3 A transaction definition points to the profile.
- 4 The profile cannot be discarded because its name begins with DFH.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

PROFILEIDERR

RESP2 values:

- 1 The profile cannot be found.

DISCARD PROGRAM

Remove a program, map set, or partition set definition.

```
►►—DISCARD PROGRAM(data-value)—◄◄
```

Conditions: INVREQ, NOTAUTH, PGMIDERR

Description

The DISCARD PROGRAM command removes the definition of a program, map set, or partition set (a load module resource) from the executing CICS system, so that the system no longer has access to the resource. That is, it revokes the earlier installation of a PROGRAM, MAPSET, or PARTITIONSET definition of the same name.

You cannot discard a module that is being executed or otherwise used by a task. Definitions supplied by CICS (modules whose name begin with DFH) and modules defined as user-replaceable (such as autoinstall programs) also are ineligible.

See “Discarding resource definitions” on page 17 for general information about discards.

Options

PROGRAM(*data-value*)

specifies the 8-character name of the program, map set, or partition set that is to be removed.

Conditions

INVREQ

RESP2 values:

- 1** The resource cannot be discarded because its name begins with DFH.
- 11** The resource definition is currently in use.
- 15** The resource cannot be discarded because it is a user-replaceable module.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.
- 101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

PGMIDERR

RESP2 values:

- 7** The resource definition cannot be found.

DISCARD TRANCLASS

Remove a transaction class definition.

```
►►—DISCARD TRANCLASS(data-value)—◄◄
```

Conditions: INVREQ, NOTAUTH, TCIDERR

Description

The DISCARD TRANCLASS command removes the definition of a transaction class from the executing CICS system.

A transaction class cannot be removed while it contains any transactions.

See “Discarding resource definitions” on page 17 for general information about discards.

Options

TRANCLASS(*data-value*)

specifies the 8-character name of the transaction class that is to be removed.

In earlier releases of CICS, transaction classes were numbered from 1 through 10 rather than named, as they are now. For compatibility, CICS supplies ten definitions that are named 'DFHTCL' followed by the 2-digit class number (DFHTCL01 for class 1, and so on). These definitions can be discarded like any other transaction class, even though they begin with DFH.

Conditions

INVREQ

RESP2 values:

- 2** The TRANCLASS definition is in use.
- 12** The transaction class cannot be discarded because installed transactions belong to it.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

TCIDERR

RESP2 values:

- 1** The transaction class cannot be found.

DISCARD TRANSACTION

DISCARD TRANSACTION

Remove a transaction definition.

►► DISCARD TRANSACTION(*data-value*)◄◄

Conditions: INVREQ, NOTAUTH, TRANSIDERR

Description

The DISCARD TRANSACTION command removes the definition of a transaction from the executing CICS system. That is, it revokes the earlier installation of a TRANSACTION resource definition of the same name.

You cannot delete transactions supplied by CICS (names beginning with the letter C), transactions defined by the CICS system initialization table (paging transactions, for example), or transactions that are scheduled to execute at a future time or when required resources are available.

See “Discarding resource definitions” on page 17 for general information about discards.

Options

TRANSACTION(*data-value*)

specifies the 4-character name of the transaction that is to be removed.

Conditions

INVREQ

RESP2 values:

- 4 The transaction cannot be discarded because its name begins with C.
- 12 The transaction definition is currently in use.
- 13 The transaction is a SIT parameter.
- 14 The transaction is scheduled to run at a future time (in use by an interval control element).
- 15 The transaction is scheduled to run when required resources are available (in use by an automatic initiate descriptor).

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.
- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

TRANSIDERR

RESP2 values:

- 1 The transaction cannot be found.

ENABLE PROGRAM

| Enable a user exit to allow it to be invoked.

►► ENABLE PROGRAM(*data-value*)◄◄

ENTRY(*ptr-ref*)
ENTRYNAME(*data-value*)
EXIT(*data-value*)
FORMATEDF
GALENGTH(*data-value*)
GAENTRYNAME(*data-value*)
LINKEDITMODE
SHUTDOWN
START
TALENGTH(*data-value*)
TASKSTART

Conditions: INVEXITREQ, NOTAUTH

Description

| The initial ENABLE PROGRAM command for an exit:

- | • Defines it as an exit to the executing CICS system and names it
- | • Sets the initial execution status (whether it is available for execution and the points at which it is executed)
- | • Allocates work areas
- | • Loads the associated load module if necessary and establishes the entry point within it for the exit.

| After the initial ENABLE command that defines the exit, you can add or remove points at which the exit is executed or change its availability dynamically with ENABLE and DISABLE commands, until you DISABLE with the EXITALL option, which deletes the definition of the exit. See the description of that command on page 53 for the correspondence between options on the two commands.

| For programming information about exits in CICS, see the *CICS/ESA Customization Guide*; you should also see the general discussion of commands that modify exits in “Exit-related commands” on page 18.

Options

ENTRY(*ptr-ref*)

| specifies a pointer reference that contains the entry point address of the global or task-related user exit. The address you specify must be within the virtual storage range occupied by the load module named in the PROGRAM option.

The use of the ENTRY option means that the module named in the PROGRAM option has already been loaded or is permanently resident. CICS does not

attempt to load the module, and also does not delete it when the user exit is disabled with EXITALL. If you omit ENTRY, CICS uses the first entry point in the load module and manages loading and deletion for you.

ENTRY is valid only on the initial ENABLE command that defines the exit.

If you specify LINKEDITMODE for a task-related user exit, the top bit of the entry address must contain the addressing mode (AMODE) indicator. The top bit is set if the exit is AMODE=31 and is zero if AMODE=24.

ENTRYNAME(*data-value*)

specifies the 8-character name of the global or task-related user exit that is to be enabled. This name must be different from the name of any exit already established. It does not have to be defined to CICS other than by means of this command, and it need not be the name of a load module or an entry point to a load module.

If you omit ENTRYNAME, the name of the exit defaults to the name of the load module specified in the PROGRAM option.

After the initial ENABLE command that defines the exit, you must use the same combination of ENTRYNAME and PROGRAM values to identify the exit on subsequent ENABLE, DISABLE, and EXTRACT EXIT commands.

EXIT(*data-value*) (**global user exits only**)

specifies the 8-character name of a global user exit point with which this exit is to be associated. When an exit is “associated” with an exit point, it is invoked when CICS reaches that particular point in its management code, provided the exit has been “started” (made available for execution). Exit points are defined and named by CICS. For programming information about exits, and a list of exit points, see the *CICS/ESA Customization Guide*.

You can name only one exit point on each ENABLE command. If the same exit is to be invoked from multiple exit points, you need a separate ENABLE command for each point.

FORMATEDF (**task-related user exits only**)

specifies that the exit is to be invoked at additional points (within EDF), when the exit is invoked by a task running under EDF. The additional invocations allow the exit to format EDF displays and interpret changes made by the user to fields on the EDF screen. You can turn off EDF invocations with a DISABLE command specifying FORMATEDF.

GAENTRYNAME(*data-value*)

specifies the 8-character name of a currently enabled global or task-related user exit whose global work area is to be shared by the exit being enabled. This is the name assigned to that exit when it was defined (its ENTRYNAME if one was used or its load module name from the PROGRAM option if not).

It must own the work area (that is, GALENGTH must have been specified when it was originally enabled).

CICS will not release a work area until all of the exits using it are disabled with EXITALL (no longer defined), but the owning exit must still be enabled for a new exit to share its work area.

GALENGTH and GAENTRYNAME are mutually exclusive and must be specified on the initial ENABLE command that defines the exit. If neither option is supplied, no global work area is provided.

GALENGTH(*data-value*)

specifies, as a halfword binary value, the length in bytes of the global work area that is to be provided by CICS for this exit. Valid lengths are 1 through 32767. The work area is initialized to binary zeros.

GALENGTH is valid only on the initial ENABLE command that defines the exit.

CICS does not return the address of the work area on the ENABLE command; you can use an EXTRACT EXIT command to determine it.

LINKEDITMODE (**task-related user exits only**)

specifies that the exit should be invoked in the addressing mode in which it was link-edited. If you do not specify LINKEDITMODE, it is invoked in the addressing mode of the caller. LINKEDITMODE is valid only on the initial ENABLE command that defines the exit.

You should avoid using LINKEDITMODE to force a task-related user exit to run in AMODE(24) because:

- An exit link-edited in AMODE(24) cannot be invoked from a task running with TASKDATALOC(ANY). If you attempt to do this, the task abends with CICS abend code AEZB.
- Enabling an exit for TASKSTART and LINKEDITMODE causes CICS to force all transactions to run with TASKDATALOC(BELOW) if the associated load module is link edited for AMODE(24).
- For a CICS shutdown call, CICS ignores the LINKEDITMODE attribute and invokes the exit in the addressing mode of the task that performs this shutdown function. For some types of shutdown, the addressing mode of this task is not predefined.

PROGRAM(*data-value*)

specifies the 8-character name of the **load module** containing the entry point of the exit. CICS uses the PROGRAM resource definition of this name to load the program, if necessary, and to verify that it is enabled and resides on the same CICS system as the exit. If no such definition exists, CICS will attempt to build one dynamically if the system is defined to allow autostall of programs.

If you omit the ENTRYNAME option, CICS assumes that the name of the exit is the same as that of the load module.

ENABLE PROGRAM

SHUTDOWN (task-related user exits only)

specifies that the exit is to be invoked during CICS shutdown processing. You can turn off the invocation with a DISABLE command specifying SHUTDOWN.

START

indicates that the exit is available for execution. You can turn availability on and off with ENABLE commands (specifying START) and DISABLE commands (specifying STOP), but the exit starts out in stopped mode and is not available until the first ENABLE with START.

When a STOPped task-related user exit gets invoked, the invoking code gets an AEY9 abend code. There is no corresponding error for global user exits, however, because CICS invokes only those exits associated with an exit point which are also available for execution (not stopped).

When a single global user exit is to be associated with several exit points, the START option allows you to delay execution of the exit until all the required ENABLE commands have been issued. You can, however, associate more exit points with the exit **after** it has been started.

TALENGTH(*data-value*) (task-related user exits only)

specifies, as a halfword binary value, the length in bytes of the work area that CICS provides for each task that uses the exit. Valid lengths are 1 through 32767. CICS allocates the work area and initializes it to binary zeros before the first use of the exit by the task, and releases it at task end. If you do not specify TALENGTH, CICS does not create task work areas.

TASKSTART (task-related user exits only)

specifies that the exit is to be invoked at the start of every task. The exit will also be invoked at end of task, but you can turn off this invocation within the exit if you wish. (The task that logs off an autoinstalled terminal in an MRO environment is an exception; it does not invoke the exit.)

The TASKSTART option is independent of the START option, but you should turn on START before or at the same time as TASKSTART, to avoid invoking the exit when it is not available for execution. In addition, you must not code the TASKSTART option on any ENABLE command that can be executed before the recovery part of CICS initialization.

You can turn off these invocations with a DISABLE command specifying TASKSTART.

Conditions

INVEXITREQ

The INVEXITREQ condition of the ENABLE command is indicated by X'80' in the first byte of EIBRCODE. The exact cause of the error can be determined by examining the second and third bytes of EIBRCODE.

X'808000' The load module named in the PROGRAM option has not been defined to CICS and could not be autoinstalled, or is not in the load library, or has been disabled, or is defined as remote, or does not contain the address specified in the ENTRY option.

X'804000' The name specified in the EXIT option is not a valid global user exit point.

X'802000' The exit is already enabled. ENTRY, LINKEDITMODE, TALENGTH, GALENGTH and GAENTRY are valid only on the initial ENABLE command that defines the exit.

X'801000' The exit is already associated with the exit point specified in the EXIT option.

X'800800' The exit specified in the GAENTRYNAME option is not enabled.

X'800400' The exit specified in the GAENTRYNAME option does not own a work area.

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

Examples

Enabling global user exits

Example 1

```
EXEC CICS ENABLE PROGRAM('EP1') ENTRYNAME('EP1')
      EXIT('XFCREQ') START
```

Example 1 defines exit EP1, tells CICS that EP1 is to be invoked from exit point XFCREQ, and makes EP1 available for execution. No global work area is obtained. CICS loads the EP module if necessary.

Example 2

```
EXEC CICS ENABLE PROGRAM('EP2') EXIT('XMNOUT')
      START ENTRY(EADDR) GALENGTH(500)
```

Example 2 defines an exit named EP2 (named by default from its load module). This module is already loaded, and the entry point for the exit is in EADDR. The exit is to be executed at exit point XMNOUT, and it is available for execution. A global work area of 500 bytes, which is to be owned by EP2, is obtained.

Example 3

```
EXEC CICS ENABLE PROGRAM('EP3') EXIT('XTDOUT')
      GAENTRYNAME('EP2')
EXEC CICS ENABLE PROGRAM('EP3') EXIT('XTDIN')
EXEC CICS ENABLE PROGRAM('EP3') EXIT('XTDREQ') START
```

| The first command of example 3 defines exit EP3; it is
 | associated with exit point XTDOUT. CICS will load module
 | EP3 if necessary. EP3 is to use the work area that is owned
 | by exit EP2. (This assumes that the ENABLE command in
 | example 2 has already been issued.)

The second command says that EP3 is also associated with
 | exit point XTDIN. The third command says that EP3 is
 | associated with exit point XTDREQ, and makes the exit
 | available for execution. EP3 is now invoked from all of these
 | exit points, and it can use EP2's work area on any of those
 | invocations.

Enabling task-related user exits

Example 1

```
EXEC CICS ENABLE PROGRAM('EP9')
        TALENGTH(750) ENTRYNAME('RM1') GALENGTH(200)

EXEC CICS ENABLE PROGRAM('EP9')
        ENTRYNAME('RM1') START
```

| The first command defines the task-related user exit RM1,
 | loads EP9 (the load module executed initially) if it is not
 | already resident, and allocates a 200-byte global work area
 | to the exit. It also schedules the allocation of a further
 | 750-byte work area for each task that invokes RM1. The
 | second command makes the exit available for execution.

EXTRACT EXIT

Obtain the address and length of a global work area.

```
▶▶—EXTRACT EXIT PROGRAM(data-value)————▶
▶—┬—ENTRYNAME(data-value)—┬—▶
▶—GALENGTH(data-area)—GASET(ptr-ref)—▶◀
```

Conditions: INVEXITREQ, NOTAUTH

Description

The EXTRACT EXIT command obtains the address and the length of the global work area that is owned by, or shared by, a user exit.

Options

ENTRYNAME(*data-value*)

| specifies the 8-character name of the global or
 | task-related user exit for which you want global work
 | area information. If you omit ENTRYNAME, CICS
 | assumes that the name of the exit is the same as the
 | name of the load module given in the PROGRAM option.
 | Therefore, you must use the same combination of
 | ENTRYNAME and PROGRAM values as was specified
 | on the ENABLE command that defined the exit.

GALENGTH(*data-area*)

| returns the length in bytes of the global work area, in
 | halfword binary form.

GASET(*ptr-ref*)

| returns the address of the global work area.

PROGRAM(*data-value*)

| specifies the name of the **load module** containing the
 | entry point of the exit. This name is also used as the
 | name of the exit when ENTRYNAME is not specified;
 | see the ENTRYNAME option.

Conditions

INVEXITREQ

| The INVEXITREQ condition of the EXTRACT EXIT
 | command is indicated by X'80' in the first byte of
 | EIBRCODE. The exact cause of the error can be
 | determined by examining the second and third bytes of
 | EIBRCODE. For further information on EIBRCODE, see
 | Appendix B, "EXEC interface block (EIB) response and
 | function codes" on page 213.

| **X'800200'** The exit is not enabled.

| **X'800400'** The exit has no global work area.

INQUIRE AUTINSTMODEL

X'808000' The load module named in the PROGRAM option is not the same as the one used when the exit specified in the ENTRYNAME option was enabled.

NOTAUTH

RESP2 values:

- | **100** The user associated with the issuing task is not authorized to use this command.
- | **101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

INQUIRE AUTINSTMODEL

Find out whether an autoinstall model is installed.

▶▶—INQUIRE AUTINSTMODEL(*data-value*)—◀◀

Conditions: END, ILLOGIC, MODELIDERR, NOTAUTH

Description

The INQUIRE AUTINSTMODEL command allows you to determine whether a particular autoinstall model is installed (defined in the current execution of your CICS system).

Browsing

You can also browse through all of the autoinstall models installed in your system by using the browse options (START, NEXT, and END) on INQUIRE AUTOINSTALL commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

AUTINSTMODEL(*data-value*)

specifies the 8-character identifier of the autoinstall model about which you are inquiring.

Conditions

END

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

MODELIDERR

RESP2 values:

- 1 The model specified cannot be found.

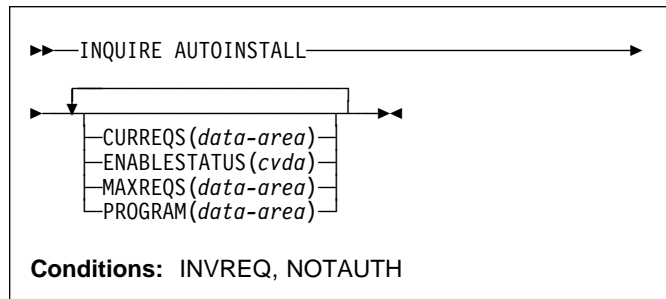
NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

INQUIRE AUTOINSTALL

Retrieve autoinstall values.



Conditions

INVREQ

RESP2 values:

- 1 VTAM is not in use in this system.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

Description

The INQUIRE AUTOINSTALL command lets you look at some of the values that control the automatic installation of VTAM terminals (autoinstall) in your CICS system.

For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Options

CURREQS(*data-area*)

returns a fullword binary field indicating the number of terminal autoinstall requests that are currently being processed. This count does not include terminals already installed in this manner.

ENABLESTATUS(*cvda*)

returns a CVDA value indicating the overall status of autoinstall for terminals. CVDA values are:

DISABLED

Terminals cannot be automatically installed.

ENABLED

Terminals can be automatically installed.

MAXREQS(*data-area*)

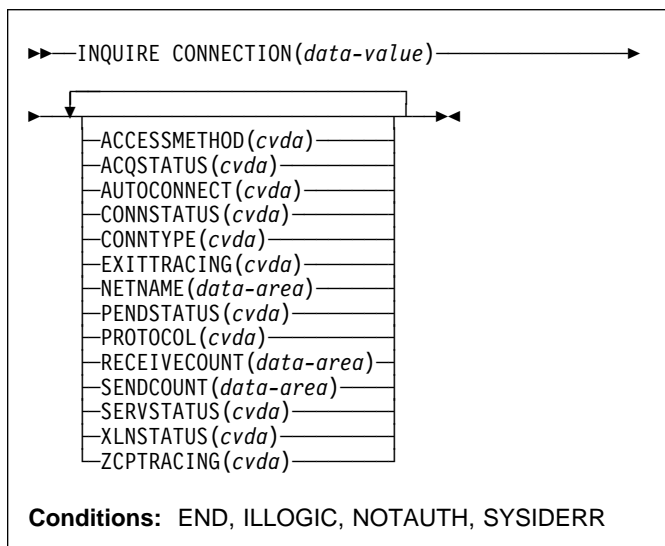
returns a fullword binary field indicating the largest number of autoinstall requests that can be processed concurrently. Note that this value has no effect on the total number of terminals that can be installed automatically.

PROGRAM(*data-area*)

returns the 8-character name of the installation-supplied program used in the autoinstall process. This is either the CICS-supplied default autoinstall program, DFHZATDX, or a user-written program.

INQUIRE CONNECTION

Retrieve information about a system connection.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE CONNECTION command retrieves information about a connection from your local CICS region to another CICS region or another system. A CONNECTION definition is sometimes known as a “system entry.”

There are two main types of system connection:

- Multiregion operation (MRO), which uses the CICS interregion communication program (DFHIRP) to establish a connection between two MRO partners. An MRO connection can exist between two CICS regions on the same MVS image or within the same MVS/ESA sysplex.

A special form of MRO connection, used by the external CICS interface (EXCI) can exist between a CICS region and a non-CICS client program running in MVS, for example an MVS batch or TSO program. An EXCI connection connects the client program to a CICS region running in the same MVS image or within the same MVS/ESA sysplex.
- Intersystem communication (ISC) connections, between CICS and any other system which supports VTAM APPC or LUTYPE6.1 communications. For example, ISC connections can exist between CICS regions running in different MVS/ESA sysplexes or on different operating system platforms, between CICS and any APPC device, and between CICS and IMS.

Browsing

You can also browse through all of the CONNECTION definitions installed in your system by using the browse options (START, NEXT and END) on INQUIRE CONNECTION commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

ACCESSMETHOD(*cvda*)

returns a CVDA value indicating the type of connection between the local system and the one you are inquiring about. CVDA values are:

INDIRECT

Communication between the local CICS system and the system defined by this connection is through the system named in the INDSYS operand of the CONNECTION definition.

IRC

The connection is used for multiregion operation (MRO), and has been defined to use DFHIRP for communication. If the CONNSTATUS is ACQUIRED, the MRO partner is running on the same MVS image. If the CONNSTATUS is RELEASED, the MRO partner may not be on the same MVS image; if not, the XCF access method will be used when the connection becomes ACQUIRED.

VTAM

The connection is used for intersystem communication (ISC).

XCF

The connection is used for multiregion operation (MRO), and communication uses the cross-system coupling facility (XCF) of MVS/ESA. XCF is used for MRO links between CICS regions on different MVS images within an MVS sysplex. It is selected dynamically by CICS for such links when the access method is defined as IRC or XM in the CONNECTION definition.

XM

The connection is used for multiregion operation (MRO) and has been defined to use MVS cross-memory (XM) services for communication. If the CONNSTATUS is ACQUIRED, the MRO partner is running on the same MVS image. If the CONNSTATUS is RELEASED, the MRO partner may not be on the same MVS image; if not, the XCF access method will be used when the connection becomes ACQUIRED.

ACQSTATUS(*cvda*) (APPC only)

returns the same value as the CONNSTATUS option and is retained only for compatibility purposes. You should use CONNSTATUS in new applications.

AUTOCONNECT(*cvda*) (VTAM only)

returns a CVDA value identifying which AUTOCONNECT option has been specified in the CONNECTION

definition. For parallel APPC connections (those with SINGLESESS(NO) specified), the AUTOCONNECT operand controls the binding of the LU services manager sessions whenever communication with VTAM is started. For single-session APPC connections and for LUTYPE6.1 connections, the AUTOCONNECT operand on the CONNECTION definition is ignored and the value returned is not meaningful. CVDA values are:

ALLCONN

AUTOCONNECT(ALL) has been specified on the CONNECTION definition. This is the same as specifying AUTOCONNECT(YES), but it can be used for consistency with the associated SESSIONS definition, which allows AUTOCONNECT(ALL).

AUTOCONN

AUTOCONNECT(YES) has been specified on the CONNECTION definition. CICS is to try to bind the LU services manager sessions.

NONAUTOCONN

AUTOCONNECT(NO) has been specified for the CONNECTION definition. CICS does not bind LU services manager sessions.

CONNECTION(*data-value*)

specifies the 4-character identifier of the remote system or region about which you are inquiring (that is, the name assigned to its CONNECTION definition).

CONNSTATUS(*cvda*) (APPC and MRO only)

returns a CVDA value identifying the state of the connection between CICS and the remote system. The remote system can be an APPC partner or a CICS MRO partner; CONNSTATUS is not applicable to EXCI or LU6.1 connections. The ACQUIRED and RELEASED CVDA values are common to both APPC and MRO; the others are unique to APPC. CVDA values are:

ACQUIRED

The connection is acquired. The criteria for ACQUIRED for VTAM links are:

- The partner LU has been contacted.
- The initial CHANGE-NUMBER-OF-SESSIONS (CNOS) exchange has been done.

The criteria for ACQUIRED for MRO links are:

- Both sides of the link are in service.
- Both sides of the link are successfully logged on to DFHIRP.
- A connection request by each side has been successful for at least one session, and therefore each side can send and receive data.

AVAILABLE

The connection is acquired but there are currently no bound sessions because they were unbound for limited resource reasons.

FREEING

The connection is being released.

OBTAINING

The connection is being acquired. The connection remains in the OBTAINING state until all the criteria for ACQUIRED have been met.

RELEASED

The connection is RELEASED. Although it may also be in INSERVICE status, it is not usable.

The RELEASED status can be caused by any one of a number of general conditions:

- The remote system has not yet initialized.
- No CONNECTION definition exists on the remote system.
- The connection on the remote system has been set out of service.

In the case of a CICS-to-CICS MRO connection, the RELEASED status may also be because:

- The remote CICS region has not yet logged on to DFHIRP.
- The remote CICS region has closed interregion communication.

In the case of an APPC ISC connection, the RELEASED status may also be because:

- The remote CICS region has not yet opened its VTAM ACB.
- AUTOCONNECT(NO) has been specified on the CONNECTION or SESSIONS definition.

NOTAPPLIC

The connection is not a CICS-to-CICS MRO connection or an APPC connection.

CONNTYPE(*cvda*) (EXCI only)

returns a CVDA value identifying the type of external CICS interface (EXCI) sessions, or pipes, defined for this connection. This option applies only to EXCI connections. CVDA values are:

GENERIC

The connection is generic. A GENERIC connection is an MRO link with many sessions to be shared by multiple users.

SPECIFIC

The connection is specific. A SPECIFIC connection is an MRO link with one or more sessions dedicated to a single user.

NOTAPPLIC

The connection is not an EXCI connection.

See the *CICS/ESA External CICS Interface* manual for more information about EXCI connections.

INQUIRE CONNECTION

EXITTRACING(*cvda*) (VTAM only)

returns a CVDA value indicating whether the terminal exit program is tracing the sessions associated with this connection. CVDA values are:

EXITTRACE

Tracing is on.

NOEXITTRACE

Tracing is off.

NOTAPPLIC

The connection is not LU6.1 or APPC.

NETNAME(*data-area*)

returns the 8-character name by which the remote system is known to the network (from the NETNAME value specified in the CONNECTION definition).

For an ISC connection, the NETNAME corresponds to the VTAM APPLID of the remote system.

For a CICS-to-CICS MRO connection, the NETNAME is the name the remote system uses to log on to DFHIRP (from the APPLID option in its system initialization table (SIT)).

For a SPECIFIC EXCI connection, NETNAME is the name of the client program which is passed on the EXCI INITIALIZE_USER command; for a GENERIC EXCI connection, NETNAME is always blanks.

For an indirect connection, NETNAME corresponds to the APPLID (as specified in the SIT APPLID option) of the terminal owning region.

PENDSTATUS(*cvda*) (APPC only)

returns a CVDA value identifying whether there are any pending units of work. CVDA values are:

NOTPENDING

There are no pending units of work.

PENDING

The local system has units of work that require resynchronization with the remote system. The resynchronization process results in CLS2 transactions being initiated when a new session is bound between the systems.

NOTAPPLIC

This is not an APPC connection.

For information on pending units of work, see the *CICS/ESA Intercommunication Guide*.

PROTOCOL(*cvda*) (VTAM and EXCI only)

returns a CVDA value identifying the protocol in use if this is a VTAM or EXCI connection. CVDA values are:

APPC

The connection uses the VTAM LUTYPE6.2 protocol for intersystem communication.

EXCI

The connection uses the external CICS interface for communication between CICS and a non-CICS client program.

LU61

The connection uses the VTAM LUTYPE6.1 protocol.

NOTAPPLIC

The connection is used for CICS-to-CICS MRO communication or it is INDIRECT.

RECEIVECOUNT(*data-area*) (MRO only)

returns a fullword binary value giving the number of RECEIVE sessions defined for this connection. This option applies only to MRO connections; for others the value returned is -1.

SENDCOUNT(*data-area*) (MRO only)

returns a fullword binary value giving the number of SEND sessions defined for this connection. For EXCI connections, the SENDCOUNT is always zero. This option applies only to MRO connections; for others the value returned is -1.

SERVSTATUS(*cvda*)

returns a CVDA value indicating whether data can be sent and received on the connection. CVDA values are:

GOINGOUT

OUTSERVICE has been requested on a SET CONNECTION command, and the request cannot be acted on until some current work has completed.

INSERVICE

Data can be sent and received.

OUTSERVICE

Data cannot be sent and received.

XLNSTATUS(*cvda*) (APPC only)

returns a CVDA value identifying the status of the exchange log names (XLN) process. CVDA values are:

NOTAPPLIC

The XLN process is not applicable. This can be because the link:

- Is released
- Is MRO, LUTYPE6.1, or single-session APPC
- Does not support synchronization level 2 conversations.

For information about the APPC exchange log names process, see the *CICS/ESA Intercommunication Guide*.

XNOTDONE

The XLN flow for the APPC connection has not completed successfully. The CSMT log can contain information relating to this state. Synchronization level 2 conversations are not allowed on the connection, but synchronization levels 0 and 1 are still allowed.

XOK

The XLN process for the APPC connection has completed successfully.

ZCPTRACING(*cvda*) (VTAM only)

returns a CVDA value indicating whether the VTAM control component of CICS is tracing activity on the sessions associated with this connection. CVDA values are:

NOTAPPLIC

The connection is not LUTYPE6.1 or APPC.

NOZCPTRACE

ZCP tracing is not active.

ZCPTRACE

ZCP tracing is active.

Conditions**END**

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

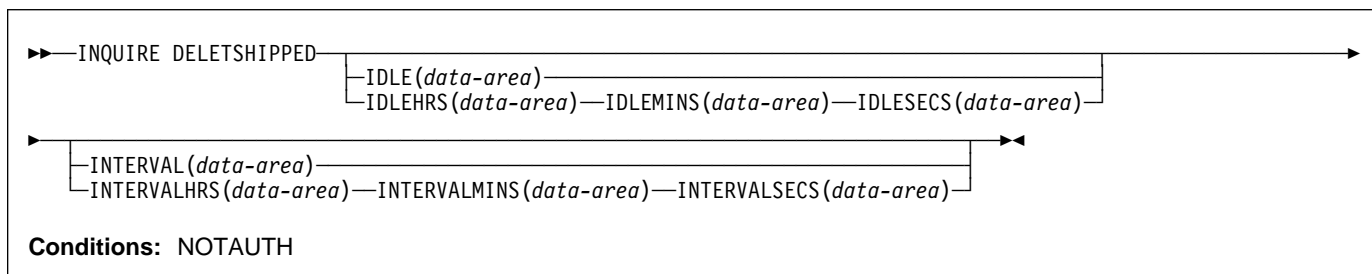
SYSIDERR

RESP2 values:

- 1 The connection cannot be found.

INQUIRE DELETSHIPED

Retrieve information about system settings that control the CICS timeout delete mechanism.



Description

CICS provides a mechanism for deleting shipped terminal definitions after they have been idle for a period of time. The installation specifies how long a terminal must have been inactive to be eligible for deletion (the IDLE time), and how often the check should be made (the INTERVAL). The INQUIRE DELETSHIPED command displays the current settings of these two control options.

There are two formats for each of the time values that you can retrieve with this command (the idle time and the interval checking period):

- A 4-byte packed decimal composite (0hhmmss+), which you obtain by using the IDLE and INTERVAL options.
- Separate hours, minutes, and seconds, which you obtain by specifying the IDLEHRS, IDLEMINS, and IDLESECS options (instead of IDLE), and INTERVALHRS, INTERVALMINS, and INTERVALSECS (instead of INTERVAL).

Options

IDLE(*data-area*)

returns the idle time, as a 4-byte packed decimal field in the format 0hhmmss+. Idle time is the minimum time that a terminal must be inactive to be eligible for deletion.

IDLEHRS(*data-area*)

returns the hours component of the idle time, in fullword binary form.

IDLEMINS(*data-area*)

returns the minutes component of the idle time, in fullword binary form.

IDLESECS(*data-area*)

returns the seconds component of the idle time, in fullword binary form.

INTERVAL(*data-area*)

returns a 4-byte packed decimal field, in the format 0hhmmss+, giving the interval at which the check for idle terminals is made.

INTERVALHRS(*data-area*)

returns the hours component of the interval, in fullword binary form.

INTERVALMINS(*data-area*)

returns the minutes component of the interval, in fullword binary form.

INTERVALSECS(*data-area*)

returns the seconds component of the interval, in fullword binary form.

Conditions

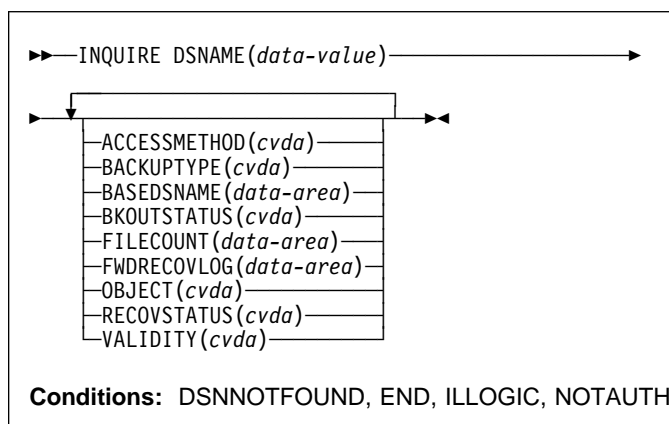
NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

INQUIRE DSNAME

Retrieve information about an external data set.



For more information about the use of CVDAs, see “CICS-value data areas (CVDA)” on page 7.

Description

The INQUIRE DSNAME command returns information about the object associated with a FILE resource definition, which can be a BDAM data set, a VSAM data set or a VSAM path to a data set through an alternate index.

Data sets are associated with files either dynamically, through the DSNAME option in the FILE definition, or statically, through the DSN option on the associated JCL DD statement. Some attributes of a data set cannot be determined until a file that references the data set has been opened. If this has not happened, UNDETERMINED values are returned for these attributes.

Browsing

You can also browse through all the objects associated with files installed in your system, by using the browse options (START, NEXT, and END) on INQUIRE DSNAME commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

ACCESSMETHOD(cvda)

returns a CVDA value identifying the access method used with this data set. CVDA values are:

BDAM

The access method is BDAM.

VSAM

The access method is VSAM.

BACKUPTYPE(cvda) (VSAM only)

returns a CVDA value identifying the type of backup used for this data set. CVDA values are:

DYNAMIC

The data set is eligible for “backup while open” (BWO) processing; that is, a data set manager with the required function can take a backup of the data set while it is open for output. The data set can also be backed up while it is closed.

NOTAPPLIC

This is a VSAM path.

STATIC

All CICS files open for output against this data set must be closed before a data set manager (such as DFHSM or DFDSS)² can take a backup copy.

UNDETERMINED

The backup type is not known because no files associated with this data set have been opened.

BASEDSNAME(data-area) (VSAM only)

returns the 44-character name of the base cluster associated with a VSAM path, when the object of the inquiry is a path. When the object is a VSAM data set, this option returns the same value as the DSNAME option.

If the access method is BDAM, blanks are returned.

BKOUTSTATUS(cvda) (VSAM only)

returns a CVDA value identifying the backout status of the data set. Backout occurs when one of the files associated with the data set is defined as recoverable and a transaction that has updated the file fails. CVDA values are:

FAILEDBKOUT

An attempt to back out has failed, and file integrity may have been lost. For further information on this topic, see the *CICS/ESA Recovery and Restart Guide*.

FAILINGBKOUT

An attempt to back out has failed, and not all associated files have completed closing.

NORMALBKOUT

All backouts have been successful, or backout has not been necessary.

² Data Facility Hierarchical Storage Manager (DFHSM) and Data Facility Data Set Services (DFDSS) are IBM licensed programs for data set backup and recovery operations.

INQUIRE DSNAME

NOTAPPLIC

This is a BDAM data set or a VSAM path. For a VSAM path, see the BKOUTSTATUS value for the associated base.

DSNAME(*data-value*)

specifies the 44-character identifier of the object about which you are inquiring. It must be associated with a FILE definition installed in CICS, named either in the DSNAME option of that definition or the JCL DD statement specified in the DDNAME option.

FILECOUNT(*data-area*)

returns a fullword binary field indicating the number of installed file definitions that refer to this data set.

FWDRECOVLOG(*data-area*) (VSAM only)

returns the number of the journal that is used to record the information required for forward recovery, in halfword binary form. If RECOVSTATUS does not have a value of FWDRECOVABLE, -1 is returned.

OBJECT(*cvda*) (VSAM only)

returns a CVDA value indicating whether the object of the inquiry is a real data set containing records (a VSAM KSDS, ESDS, or RRDS, or an alternate index used directly) or a VSAM path definition that links an alternate index to its base cluster. CVDA values are:

BASE

This is a data set containing records.

PATH

This is a path.

RECOVSTATUS(*cvda*)

returns a CVDA value identifying the recovery characteristics of the data set. CVDA values are:

FWDRECOVABLE

All updates to the data set are logged for both backout and forward recovery.

NOTAPPLIC

This is a BDAM data set or a VSAM path.

NOTRECOVABLE

Updates to the data set are not logged.

This response may also be returned as the result of use of the XFCNREC global user exit. A program enabled at XFCNREC may indicate that file opens should proceed even if there is a mismatch in the backout recovery requirements for different files associated with same data set. In these circumstances, the data set is marked as NOTRECOVABLE to indicate that its data integrity can no longer be guaranteed.

The condition remains until cleared by a CEMT SET DSNAME REMOVE, EXEC CICS SET DSNAME REMOVE or a cold start (if the associated data set is not in backout-failed state). While the data set is in this state, backout logging is performed for a particular request based on the

specification in the file definition. Therefore backout logging may occur for requests via one file and not via another.

RECOVERABLE

All updates to the data set are logged for backout.

UNDETERMINED

The recovery status is unknown, because no file associated with this data set has been opened.

VALIDITY(*cvda*)

returns a CVDA value identifying whether the data set name has been validated against the VSAM catalog by opening a file associated with the data set. CVDA values are:

INVALID

The data set name has not been validated (validation has not yet occurred or has failed).

VALID

The data set name has been validated.

You cannot find out what the RECOVSTATUS of a data set is unless VALIDITY has a setting of VALID.

Conditions

DSNNOTFOUND

RESP2 values:

- 1 The data set cannot be found.

END

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

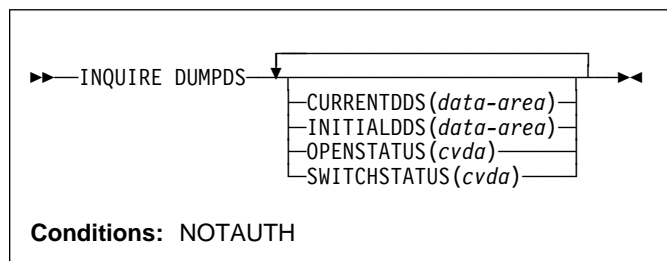
NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

INQUIRE DUMPDS

Retrieve information about the CICS transaction dump data sets.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE DUMPDS command allows you to retrieve information about CICS transaction dump data sets. There can either be one of these, known as the ‘A’ data set, or two: ‘A’ and ‘B’. One is “active” (receiving dumps) and the other, if there are two, is “inactive” (standby).

Options

CURRENTDDS(*data-area*)

returns the 1-character designator of the active dump data set (A or B). The active dump data set is not necessarily open.

INITIALDDS(*data-area*)

returns a 1-character value indicating which dump data set CICS designates as active at startup.

- A Dump data set A is active initially.
- B Dump data set B is active initially.
- X The dump data set that was not active when CICS last terminated (normally or abnormally) is active initially.

OPENSTATUS(*cvda*)

returns a CVDA value identifying the status of the active CICS dump data set. CVDA values are:

CLOSED

The active CICS dump data set is closed.

OPEN

The active CICS dump data set is open.

SWITCHSTATUS(*cvda*)

returns a CVDA value indicating whether CICS should switch active data sets when the current one fills. CVDA values are:

NOSWITCH

No automatic switching occurs.

SWITCHNEXT

When the data set designated as active at startup fills, CICS closes it, opens the other, and makes that one active. This automatic switch occurs only once, when the first active data set fills; thereafter, switching is under manual or program control.

Conditions

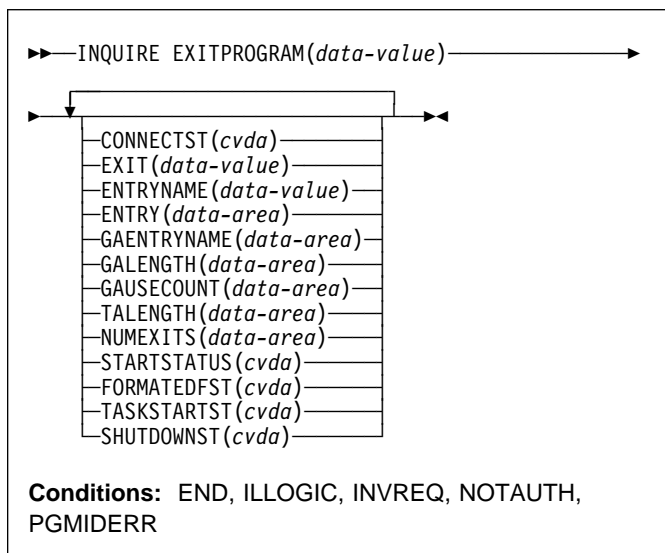
NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

INQUIRE EXITPROGRAM

Retrieve information about a user exit.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE EXITPROGRAM command returns information about a global or task-related user exit. You identify the exit about which you are inquiring with the ENTRYNAME and EXITPROGRAM options.

Browsing

You can also browse through the exit definitions in two different ways. To look at all of the global user exits defined at a particular exit point, you specify the exit point on the command that starts the browse, thus:

```

Browse EXITPROGRAM
INQUIRE EXITPROGRAM EXIT(data-value) START
    
```

To look at all user exits, both global and task-related, you omit the EXIT option on the command that starts the browse. You can distinguish between the two types by looking at the NUMEXITS value, which will be zero for a task-related exit and positive for a global exit.

On either type of browse, the sequence in which the exits are retrieved is the time order in which they were enabled.

Options

CONNECTST(cvda)

returns a CVDA value indicating the state of the connection between the exit and the external resource manager that it supports. This option applies only to the task-related user exits that connect CICS to DBCTL or to DB2. It enables you to determine whether the specified exit has connected to its resource manager, so that CICS tasks can safely issue API requests to the resource manager.

To inquire about the connection to DBCTL, use an EXITPROGRAM value of DFHDBAT and an ENTRYNAME value of DBCTL.

To inquire about the connection to DB2, use an EXITPROGRAM value of DSN2EXT1, with an ENTRYNAME of DSNCSQL, DSNCCMD, or DSNCIFC.

CVDA values are:

CONNECTED

The task-related user exit is connected to its external resource manager subsystem, and API requests can be issued.

NOTAPPLIC

The exit is not a DBCTL or DB2 task-related user exit.

NOTCONNECTED

The task-related user exit is not connected to its external resource manager subsystem, and therefore API requests cannot be issued.

If the DBCTL or DB2 task-related user exits are not enabled, the INQUIRE command returns PGMIDERR. This also indicates that CICS is not connected to one of these resource manager subsystems.

Note: To determine whether DB2 or DBCTL is available, use CONNECTST rather than STARTSTATUS, because the task-related user exit can be started without having succeeded in making its database manager available to CICS.

ENTRY(data-area)

returns a fullword binary field indicating the entry address of the user exit.

ENTRYNAME(data-value)

specifies the 8-character name of the exit about which you are inquiring. If you omit ENTRYNAME, CICS assumes that the name of the exit is the same as the name of the load module specified in the EXITPROGRAM option. Consequently, you must specify the same values for ENTRYNAME and EXITPROGRAM as were specified in the ENTRYNAME and PROGRAM options on the ENABLE command that created the exit. (EXITPROGRAM in this command corresponds to PROGRAM in an ENABLE command.)

EXIT(data-value) (global user exits only)

specifies the 8-character identifier of an exit point with which the exit about which you are inquiring is associated. You must specify an exit point when you

inquire about a global user exit. Exit points do not apply to task-related user exits, however, and you must not specify this option when you inquire about such an exit.

EXITPROGRAM(*data-value*)

specifies the 8-character name of the load module associated with the exit about which you want information. This is the value that was specified in the PROGRAM option of the ENABLE command that defined the exit.

FORMATEDFST(*cvda*) (task-related user exits only)

returns a CVDA value indicating that the FORMATEDF option is enabled for the exit. FORMATEDF causes extra invocations of the exit for tasks executed under EDF, to format output screens and interpret input, and applies only to task-related user exits. CVDA values are:

FORMATEDF

FORMATEDF is turned on.

NOFORMATEDF

FORMATEDF processing is turned off.

NOTAPPLIC

This is a global user exit.

GAENTRYNAME(*data-area*)

returns the 8-character name of the user exit that owns the global work area used by the exit about which you are inquiring.

This value is returned only when the exit uses a global work area owned by another exit. Blanks are returned if it has allocated its own work area.

GALENGTH(*data-area*)

returns a halfword binary field indicating the length of the global work area for the exit.

GAUSECOUNT(*data-area*)

returns a halfword binary field indicating the total number of global or task-related user exits that are using the global work area owned by this exit. This count includes the owning exit program. A zero is returned if the exit is not the owner.

NUMEXITS(*data-area*) (global user exits only)

returns a halfword binary field indicating the number of global user exit points at which the exit is enabled. A zero is returned if this is a task-related user exit.

SHUTDOWNST(*cvda*) (task-related user exits only)

returns a CVDA value indicating whether the SHUTDOWN option is enabled for the exit. SHUTDOWN causes invocation during CICS shutdown, and applies only to task-related user exits. CVDA values are:

NOSHUTDOWN

The exit is not invoked when a CICS shutdown occurs.

NOTAPPLIC

This is a global user exit.

SHUTDOWN

The exit is invoked when a CICS shutdown occurs.

STARTSTATUS(*cvda*)

returns a CVDA value identifying whether the exit is available for execution. CVDA values are:

STARTED

The exit program is available for execution; that is, the START option on an EXEC CICS ENABLE command is still in force.

STOPPED

The exit program is not available for execution; that is, the START option has not been issued, or has been revoked by the STOP option on an EXEC CICS DISABLE command.

TALENGTH(*data-area*) (task-related user exits only)

returns a halfword binary field indicating the length of the local (task-related) work area for the exit. Local work areas apply only to task-related user exits. A zero is returned if this is a global user exit.

TASKSTARTST(*cvda*) (task-related user exits only)

returns a CVDA value indicating whether the TASKSTART option is enabled for the exit. TASKSTART causes CICS to invoke the exit at the start and end of every task; it applies only to task-related user exits. CVDA values are:

NOTAPPLIC

This is a global user exit.

NOTASKSTART

The exit is not set for invocation at the start and end of every task.

TASKSTART

The exit is set for invocation at the start and end of every task.

Conditions

END

RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

INVREQ

RESP2 values:

INQUIRE FILE

3 The exit point identified by EXIT does not exist.

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

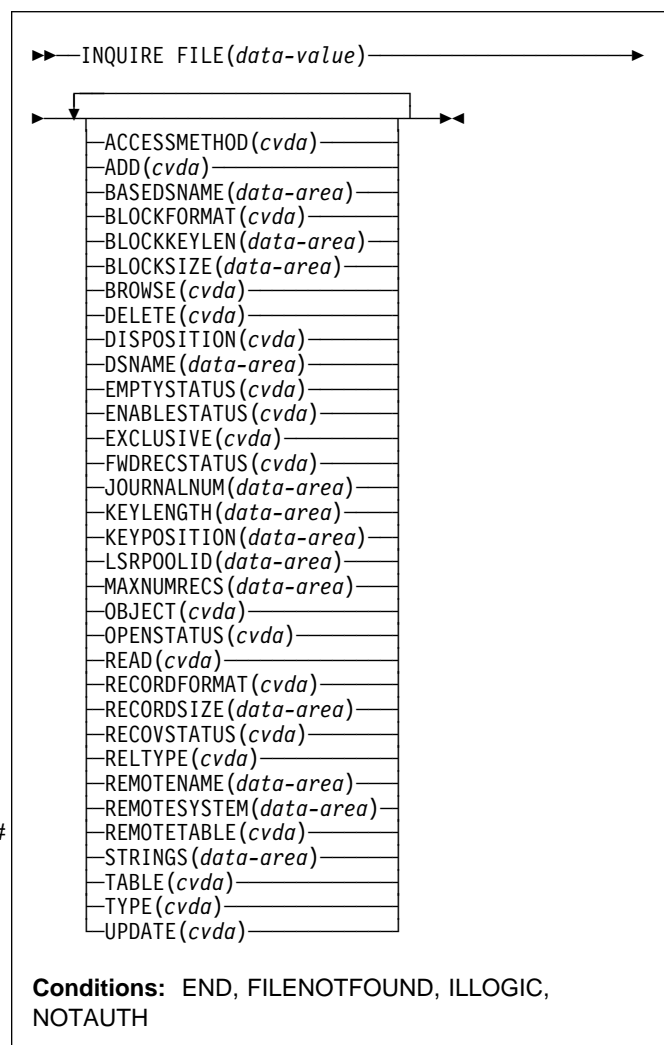
PGMIDERR

RESP2 values:

1 The exit identified by EXITPROGRAM and ENTRYNAME is not enabled, or the EXIT parameter is missing on an inquiry on a global user exit, or is present on a task-related user exit.

INQUIRE FILE

Retrieve information about a file.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE FILE command returns information about a FILE resource definition. When the file is associated with a VSAM or BDAM object, INQUIRE FILE returns information about the associated object as well. For VSAM, the object may be a base cluster (a KSDS, ESDS, or RRDS), an alternate index, or a path to a base cluster through an alternate index; for BDAM, the object is a single MVS BDAM data set. (You cannot use INQUIRE FILE to get information about DL/I data sets or data sets associated with other CICS resources or functions. However, see the INQUIRE DUMPDS, JOURNALNUM, TDQUEUE, and VOLUME commands if you need such information.)

The values returned depend on whether the file is open or closed and, if it is closed, whether it ever has been open during the current execution of CICS. If the file is not open, you get default or null values, or values describing the most recent object associated with the file, as noted in the option descriptions that follow.

The values returned also depend on whether the file is local (defined on the same CICS system as the task making the inquiry) or remote (defined on another CICS system). Less information is available for remote files, and so defaults or nulls are returned for some options.

For further information about null values, see “Command format” on page 4.

+ Also, if a file is empty (in VSAM load mode), after the first + write or massinsert has completed the file is closed and left + enabled. It remains so until the next access (write or read) + when it is implicitly opened. If an INQUIRE is done on the + file before this next access occurs, the file shows + CLOSED,ENABLED. This can be a temporary state for a file + that has just come out of load mode status.

Note: INQUIRE FILE replaces the INQUIRE DATASET command of earlier CICS releases. The translator still accepts DATASET as a synonym for FILE in this command, along with option names that have changed, but you should use FILE and the new option names in new applications.

Browsing

You can also browse through all of the files installed in your system by using the browse options (START, NEXT, and END) on INQUIRE FILE commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

ACCESSMETHOD(*cvda*)

returns a CVDA value identifying the access method for this file. CVDA values are:

BDAM

The access method is BDAM.

REMOTE

The file is defined as remote, and therefore the access method is not known to the local CICS system.

VSAM

The access method is VSAM.

ADD(*cvda*)

returns a CVDA value identifying whether new records can be added to the file. CVDA values are:

ADDABLE

New records can be added to the file.

NOTADDABLE

New records cannot be added to the file.

BASEDSNAME(*data-area*) (VSAM only)

returns the 44-character name of the base cluster associated with a VSAM path, if the object associated with the file is a path. If the object is other than a path, this option returns the same value as the DSNAME option.

Note: The translator still accepts BASENAME for this option, but you should use BASEDSNAME in new code.

BLOCKFORMAT(*cvda*) (BDAM only)

returns a CVDA value identifying whether records on the file are blocked or unblocked. CVDA values are:

BLOCKED

The records on the file are blocked, or this is a VSAM file.

UNBLOCKED

The records on the file are unblocked.

BLOCKKEYLEN(*data-area*) (BDAM only)

returns a fullword binary field indicating the physical block key length for the file.

BLOCKSIZE(*data-area*) (BDAM only)

returns a fullword binary field indicating the length in bytes of a block. If the blocks are of variable length or are undefined, the value returned is the maximum.

BROWSE(*cvda*)

returns a CVDA value identifying whether you can browse the file. CVDA values are:

BROWSABLE

You can browse the file.

NOTBROWSABLE

You cannot browse the file.

DELETE(*cvda*) (VSAM only)

returns a CVDA value identifying whether you can delete records from the file. CVDA values are:

DELETABLE

You can delete records from the file.

NOTDELETABLE

You cannot delete records from the file.

DISPOSITION(*cvda*)

returns a CVDA value indicating the value of the DISPOSITION option for the file (from the DISPOSITION option in the FILE definition or the JCL DD statement to which it points). CVDA values are:

OLD Disposition is OLD.

SHARE

Disposition is SHARE.

DSNAME(*data-area*)

returns the 44-character name of the BDAM data set or VSAM object associated with the FILE definition.

INQUIRE FILE

EMPTYSTATUS(*cvda*) (VSAM only)

returns a CVDA value indicating whether the FILE definition specifies the EMPTYREQ option. EMPTYREQ causes the object associated with this file to be set to empty, if eligible, when the file is opened. VSAM data sets defined as reusable are the only ones that you can make empty in this way; EMPTYREQ has no effect on other objects. CVDA values are:

EMPTYREQ

The data set should be made empty.

NOEMPTYREQ

The data set should not be made empty.

ENABLESTATUS(*cvda*)

returns a CVDA value identifying whether application programs can access the file. CVDA values are:

DISABLED

The file is unavailable for access by application programs because it has been explicitly disabled. It must be explicitly enabled by an EXEC CICS SET FILE ENABLED command or its CEMT equivalent before it can be accessed by application programs.

DISABLING

A request to disable the file has been received, but tasks are executing that had previously accessed the file. These tasks are allowed to complete their use of the file, but new tasks are not allowed access.

ENABLED

The file is available for access by application programs.

UNENABLED

The file is unavailable for access by application programs because it is closed. It must be explicitly enabled by an EXEC CICS SET FILE OPEN command or its CEMT equivalent before it can be accessed by application programs.

UNENABLING

A request to close the file has been received, but
tasks are executing that had previously accessed
the file. These tasks are allowed to complete their
use of the file, but new tasks are not allowed
access.

EXCLUSIVE(*cvda*) (BDAM only)

returns a CVDA value identifying whether records on this file are to be placed under exclusive control when a read for update is issued. CVDA values are:

EXCTL

A record on this file is placed under exclusive control of the reading task when it is read for update.

NOEXCTL

A record on this file is not placed under exclusive control when it is read for update.

FILE(*data-value*)

specifies the 8-character name of the file about which you are inquiring.

FWDRECSTATUS(*cvda*) (VSAM only)

returns a CVDA value identifying whether the file is forward-recoverable. CVDA values are:

FWDRECOVABLE

The file is forward-recoverable. The RECOVERY option of the FILE definition specifies that updates to the file will be recorded, to make forward recovery of the file possible.

NOTFWDRCVBLE

The file is not forward-recoverable.

JOURNALNUM(*data-area*)

+ returns a halfword binary field indicating the number of
+ the journal on which CICS writes the information
+ required for automatic journaling (the JOURNAL option
+ value in the FILE definition). Journal numbers are
+ between 1 and 99; '1' identifies the system log. A value
+ of zero indicates no journal activity for this file
+ (JOURNAL(NO) in the file definition).

KEYLENGTH(*data-area*)

returns a fullword binary field indicating the length of the key if the file is associated with a VSAM KSDS, or the length of the logical key used for deblocking if the file is associated with a BDAM data set. If there is no key, zero is returned.

KEYPOSITION(*data-area*)

returns a fullword binary field indicating the starting position of the key field in each record relative to the beginning of the record. The start is made at position 0. If there is no key, zero is returned.

LSRPOOLID(*data-area*) (VSAM only)

returns a fullword binary field indicating the number of the VSAM LSR pool associated with this file, in the range 1–8. If the file does not share buffers, this value is 0.

MAXNUMRECS(*data-area*) (VSAM only)

returns a fullword binary field indicating the maximum number of records that the data table for this file can hold. A value of zero is returned if the file is not a data table (that is, if no limit was specified in the corresponding option of the FILE definition, or if the file is remote).

OBJECT(*cvda*) (VSAM only)

returns a CVDA value indicating whether the file is associated with a real data set containing records (a VSAM KSDS, ESDS, or RRDS, or an alternate index used directly) or a VSAM path that links an alternate index to its base cluster. CVDA values are:

BASE

The file is associated with a real data set containing records.

PATH

The file is associated with a path.

You get a value of PATH only if the file defines a path to a VSAM base data set through an alternate index. If the file definition allows direct access to an alternate index, or if the path is used merely as an alias to a base data set, you get a value of BASE.

If the file is a data table, the OBJECT option refers to its source data set.

OPENSTATUS(*cvda*)

returns a CVDA value identifying whether the file is open, closed, or in a transitional state. The OPENSTATUS value affects the ability of application tasks to access the file, but only indirectly; see the ENABLESTATUS option description for the rules. CVDA values are:

CLOSED

The file is closed.

CLOSING

The file is in the process of being closed. Closing a file may require dynamic deallocation of data sets and deletion of shared resources, in which case close processing may last a significant length of time.

CLOSEREQUEST

The file is open and in use by one or more application tasks. An EXEC CICS SET FILE CLOSED or a CEMT SET FILE CLOSED request has been received, but closing is not complete (the ENABLESTATUS of the file is DISABLING).

OPEN

The file is open.

OPENING

The file is in the process of being opened.

READ(*cvda*)

returns a CVDA value identifying whether you can read records from the file. CVDA values are:

NOTREADABLE

You cannot read records from the file.

READABLE

You can read records from the file.

RECORDFORMAT(*cvda*)

returns a CVDA value identifying the format of the records on the file. CVDA values are:

FIXED

The records are of fixed length.

UNDEFINED

The format of records on the file is undefined. The UNDEFINED value is possible for BDAM data sets only.

VARIABLE

The records are of variable length. If the file is associated with a data table, the record format is always variable length, even if the source data set contains fixed-length records.

RECORDSIZE(*data-area*)

returns a fullword binary field indicating the actual size of fixed-length records, and the maximum size of variable-length records.

RECOVSTATUS(*cvda*)

returns a CVDA value identifying whether the file is recoverable. CVDA values are:

NOTRECOVERABLE

The file is not recoverable.

RECOVERABLE

The file is recoverable.

RELTYPE(*cvda*) (BDAM only)

returns a CVDA value indicating whether relative or absolute addressing is used to access the file and, if relative, the type of relative addressing. CVDA values are:

BLK Relative block addressing is being used.

DEC The zoned decimal format is being used.

HEX The hexadecimal relative track and record format is being used.

NOTAPPLIC

Absolute (MBBCHHR) addressing is being used (or the file is a VSAM file).

REMOTENAME(*data-area*)

returns the 8-character name by which the file is known in the CICS region named in the REMOTESYSTEM option of its FILE definition. Blanks are returned if the file is not remote.

REMOTESYSTEM(*data-area*)

returns a 4-character name of the CICS region in which the file is defined (from the REMOTESYSTEM value in the FILE definition). Blanks are returned if the file is not remote.

#

APAR PQ18025

#

added REMOTETABLE

REMOTETABLE(*cvda*) (VSAM only)

returns a CVDA value indicating whether the file represents an open remote data table. CVDA values are:

REMTABLE

The file represents an open remote data table.

STRINGS(*data-area*) (VSAM only)

returns a fullword binary field indicating the number of strings (concurrent operations) specified for the file in its FILE definition.

INQUIRE FILE

TABLE(*cvda*) (VSAM only)

returns a CVDA value indicating whether the file represents a data table. CVDA values are:

CICSTABLE

The file represents a CICS-maintained data table.

NOTTABLE

The file does not represent a data table.

USERTABLE

The file represents a user-maintained data table.

TYPE(*cvda*)

returns a CVDA value identifying the type of data set that corresponds to this file. CVDA values are:

ESDS

The data set is an entry-sequenced data set.

KEYED

The data set is addressed by physical keys.

KSDS

The data set is a key-sequenced data set.

NOTKEYED

The data set is not addressed by physical keys.

RRDS

The data set is a relative record data set.

UPDATE(*cvda*)

returns a CVDA value identifying whether the file is updatable. CVDA values are:

NOTUPDATABLE

You cannot update records.

UPDATABLE

You can update records.

- 100** The user associated with the issuing task is not authorized to use this command.
- 101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

Conditions

END

RESP2 values:

- 2** There are no more resource definitions of this type.

FILENOTFOUND

RESP2 values:

- 1** The file cannot be found.

ILLOGIC

RESP2 values:

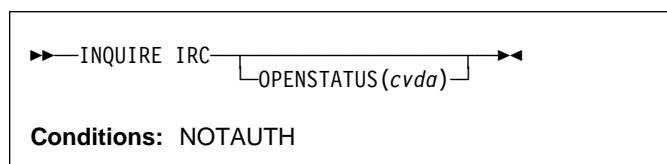
- 1** You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

INQUIRE IRC

Show the IRC status.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE IRC command indicates whether interregion communication (IRC) is open, closed, or in a transitional state in your CICS system. IRC must be open for your CICS region to communicate with another CICS region using any of the multiregion operation (MRO) facilities (IRC, XM, or XCF).

Options

OPENSTATUS(*cvda*)

returns a CVDA value identifying the status of IRC in the system. CVDA values are:

CLOSED

IRC is closed for this system, or is not present in the system.

CLOSING

A SET IRC CLOSED request to quiesce MRO has been received; tasks that were already using an MRO link are being allowed to complete, but new tasks cannot use an MRO link.

IMMCLOSING

A SET IRC IMMCLOSE request to shut down MRO immediately has been received. Tasks that were using an MRO link are being terminated abnormally.

OPEN

IRC is open for this system.

Conditions

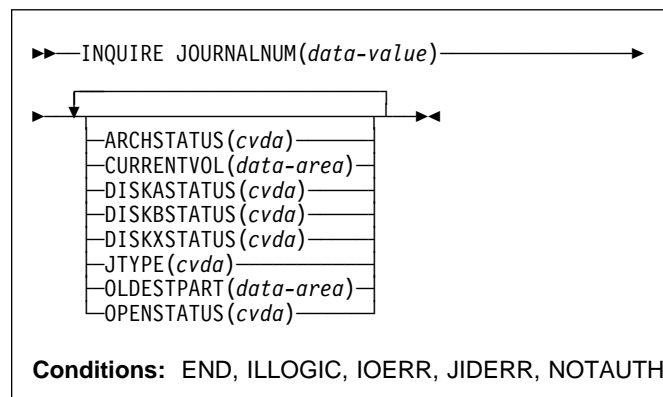
NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

INQUIRE JOURNALNUM

Retrieve information about the CICS system log or a user journal.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE JOURNALNUM command allows you to inquire about the attributes of a journal defined in the CICS journal control table (JCT), including the system log (Journal 1).

Browsing

You can also browse through all of the journals defined in your system by using the browse options (START, NEXT, and END) on INQUIRE JOURNALNUM commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

ARCHSTATUS(*cvda*)

returns a CVDA value indicating whether the journal is subject to automatic archiving. Automatic archiving occurs when the journal definition includes AUTOARCH among the JOUROPT values, and is available only for journals consisting of two disk data sets (journals defined with a JTYPE value of DISK2). CVDA values are:

AUTOARCH

The journal is subject to automatic archiving.

NOAUTOARCH

The journal is not subject to automatic archiving.

REVERTED

Automatic archiving was requested, but an error has caused a reversion to operator archiving for this journal. That is, the journal was defined with

INQUIRE JOURNALNUM

a JOUROPT value of AUTOARCH, but is currently being handled as if PAUSE had been specified instead, to prevent reuse of a journal data set before it has been archived successfully.

CURRENTVOL(*data-area*)

returns the 6-character identifier of the volume that is currently being written. CURRENTVOL applies to tape journals only; for other journals, this value is blanks.

DISKASTATUS(*cvda*)

returns a CVDA value indicating the status of the first ('A') or only data set of a disk journal. CVDA values are:

CURRENT

The data set is the current journal data set.

NOTAPPLIC

The journal is not a disk journal.

NOTREADY

The data set is waiting to be archived.

READY

The data set is ready to be written.

DISKBSTATUS(*cvda*)

returns a CVDA value indicating the status of the second ('B') data set of a journal consisting of two disk data sets. CVDA values are:

CURRENT

The data set is the current journal data set.

NOTAPPLIC

The journal is not a disk journal or its JTYPE value is not DISK2.

NOTREADY

The data set is waiting to be archived.

READY

The data set is ready to be written.

DISKXSTATUS(*cvda*)

returns a CVDA value indicating the status of the extra ('X') data set of a journal consisting of two disk data sets. The CVDA values are the same as for the DISKBSTATUS option.

JOURNALNUM(*data-value*)

specifies the number of the journal about which you are inquiring, in halfword binary form. Journal 1 is the system log.

JTYPE(*cvda*)

returns a CVDA value identifying what type of journal this is. CVDA values are:

DISK1

The journal is written to a single, reusable data set on disk.

DISK2

The journal is written to two data sets on disk ('A' and 'B'). CICS writes serially to each data set, switching automatically to the other when one is

full. DISK2 journals are defined with a JTYPE value of DISK2 and JOUROPT values including AUTOARCH.

DISK2PAUSE

This is the same as DISK2, except that CICS will not reuse a data set that has already been written without operator verification that the current data is no longer needed.

DISK2PAUSE describes a journal defined with a JTYPE value of DISK2 and PAUSE among the JOUROPT values.

SMF The journal data is written to an SMF data set.

TAPE1

The journal is written to a single tape data set.

TAPE2

The journal is written to two tape data sets. CICS writes serially to each data set, switching automatically to the other when one is full, as it does for DISK2 journals. The alternate tape data set can be placed on a second drive ready for use when the first data set fills.

OLDESTPART(*data-area*)

returns a fullword binary field giving the serial number of the first volume in the cycle or series associated with this journal. OLDESTPART applies to tape journals only; for other types, the value returned is -1.

OPENSTATUS(*cvda*)

returns a CVDA value indicating whether the journal is open for output. CVDA values are:

CLOSED

The journal is not open for output. CLOSED includes transitional states.

OPENOUTPUT

The journal is open for output.

Conditions

END

RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

IOERR

RESP2 values:

5 An error has occurred in the device, the medium, or the access method.

- 7 An error occurred writing the journal archiving control data set (JACD).
- 8 A logic error occurred in recording an archive in the journal archiving control data set (JACD).

JIDERR

RESP2 values:

- 1 The journal cannot be found.

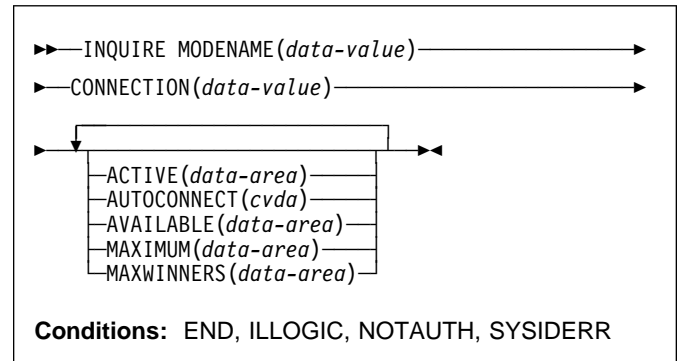
NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.
- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

INQUIRE MODENAME

Retrieve information about a session group within a connection.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE MODENAME command returns information about a group of sessions (sometimes called a “mode”) that has been defined within a connection to a remote system. (The MODENAME for the group is the name assigned to the SESSIONS resource definition that creates it.)

MODENAMES are unique within a given connection, but not across connections. Therefore, to look at a particular session group, you must specify data values for both the MODENAME and CONNECTION options.

Browsing

You can also browse through all of the session groups for a particular connection, or all groups for all connections, by using the browse options (START, NEXT, and END) on INQUIRE MODENAME commands.

As in a single INQUIRE MODENAME command, you must include both the MODENAME and CONNECTION options on an INQUIRE MODENAME NEXT command. The data-area for MODENAME is optional; if you provide it, CICS uses it to return the name of the session group. The data-area for CONNECTION is required, however. If you want to limit your browse to a single connection, specify its name there. To see all groups, initialize this value to nulls on **each** INQUIRE MODENAME NEXT command, and CICS will use the data-area to return the connection name.

See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

INQUIRE MODENAME

Options

ACTIVE(*data-area*)

returns a halfword binary field giving the number of sessions within the group which are currently in use.

AUTOCONNECT(*cvda*)

returns a CVDA value indicating whether the sessions within this group are to be bound automatically whenever CICS starts communication with VTAM. CVDA values are:

ALLCONN

CICS tries to bind both contention-winner and contention-loser sessions.

AUTOCONN

CICS tries to bind only sessions for which it is contention winner.

NONAUTOCONN

CICS does not try to bind any sessions.

AVAILABLE(*data-area*)

returns a halfword binary field giving the current number of sessions in the group (the number "bound").

CONNECTION(*data-value*)

specifies the 4-character identifier of the remote system with which this group of sessions is associated (the name of the CONNECTION resource definition for that system).

MAXIMUM(*data-area*)

returns a halfword binary field giving the maximum number of sessions that the definition of the session group permits.

MAXWINNERS(*data-area*)

returns a halfword binary field giving the maximum number of sessions that the definition of the session group permits to be contention winners. A single-session APPC definition installed by RDO or autoinstall always shows 0 for this field.

MODENAME(*data-value*)

specifies the 8-character identifier of the group of sessions about which you are inquiring. This is the name of the SESSIONS resource definition for the group.

Conditions

END

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END

command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

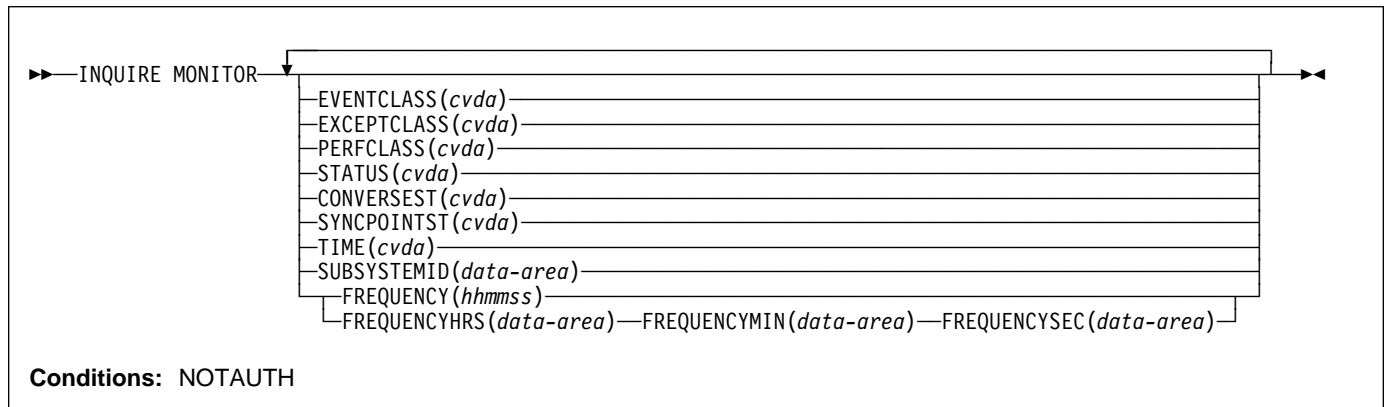
SYSIDERR

RESP2 values:

- 1 The connection cannot be found.
- 2 The modename within the connection cannot be found.
- 3 The connection specified on an INQUIRE MODENAME NEXT cannot be found.

INQUIRE MONITOR

Retrieve the status of CICS monitoring.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

The INQUIRE MONITOR command allows you to find out whether CICS monitoring is active, what types of data are being recorded, and other recording options.

CICS monitoring is controlled by a master switch (the STATUS option) and three switches that govern what types of data are recorded (the EVENTCLASS, EXCEPTCLASS, and PERFCLASS options). See the SET MONITOR command on page 166 for a description of monitor data classes and details of how the switches interact.

Options

CONVERSEST(*cvda*)

returns a CVDA value indicating how CICS is to record performance data for conversational tasks (tasks that wait for terminal or session input). CVDA values are:

CONVERSE

CICS produces a performance class record for a conversational task each time it waits for terminal input as well as at task end, representing the part of the task since the previous terminal wait (or task start). These waits occur during execution of a CONVERSE command or a RECEIVE command that follows a SEND.

NOCONVERSE

CICS accumulates performance data across terminal waits and produces a single performance class record for a conversational task.

EVENTCLASS(*cvda*)

returns a CVDA value indicating whether the SYSEVENT class of monitoring data is recorded when monitoring is active. CVDA values are:

EVENT

SYSEVENT data is recorded.

NOEVENT

SYSEVENT data is not recorded.

EXCEPTCLASS(*cvda*)

returns a CVDA value indicating whether the exception class of monitoring data is recorded when monitoring is active. CVDA values are:

EXCEPT

Exception data is recorded.

NOEXCEPT

Exception data is not recorded.

FREQUENCY(*hhmmss*)

returns the interval at which CICS produces performance class records for long-running tasks. If a task runs longer than the FREQUENCY interval, CICS records its performance data separately for each interval or fraction.

There are two formats for the frequency interval:

- A composite (packed decimal format 0hhmmss+, 4 bytes long) which you obtain by using the FREQUENCY option.
- Separate hours, minutes, and seconds, which you obtain by specifying the FREQUENCYHRS, FREQUENCYMIN, and FREQUENCYSEC options.

(A value of zero means that frequency reporting is inactive; that is, recording of performance data is not affected by the duration of the task.)

FREQUENCYHRS(*data-area*)

returns the hours component of the frequency interval, in fullword binary form (see the FREQUENCY option).

FREQUENCYMIN(*data-area*)

returns the minutes component of the frequency interval, in fullword binary form (see the FREQUENCY option).

INQUIRE MONITOR

FREQUENCYSEC(*data-area*)

returns the seconds component of the frequency interval, in fullword binary form (see the FREQUENCY option).

GMT Time stamps are Greenwich mean time.

LOCAL

Time stamps are local time.

PERFCLASS(*cvda*)

returns a CVDA value indicating whether the performance class of monitoring data is recorded when monitoring is active. CVDA values are:

NOPERF

Performance data is not recorded.

PERF

Performance data is recorded.

Conditions

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

STATUS(*cvda*)

returns a CVDA value identifying whether CICS monitoring is active in the system. CVDA values are:

OFF CICS monitoring is not active in the system. No monitoring data is accumulated or written out, irrespective of the settings of the monitoring data classes.

ON CICS monitoring is active. Data is accumulated for all classes of monitor data, and written out for those classes that are active.

SUBSYSTEMID(*data-area*)

returns the 4-character name used as the subsystem identification in SYSEVENT class records. For information on the SYSEVENT class of monitoring data and the subsystem identification, see the *CICS/ESA Performance Guide*. For information on how to specify the subsystem identification, see the MNSUBSYS parameter in the *CICS/ESA Resource Definition Guide*.

SYNCPOINTST(*cvda*)

returns a CVDA value indicating whether CICS records performance class data separately for each logical unit of work (LUW) within tasks that contain multiple LUWs. An LUW within a task ends when a syncpoint occurs, either explicitly (a SYNCPOINT command) or implicitly (a DL/I TERM call, for example, or task end); a new LUW begins immediately after, except at end of task. When rollback occurs on a syncpoint, the LUW does not end. CVDA values are:

NOSYNCPOINT

Performance data is combined over all LUWs in a task for recording.

SYNCPOINT

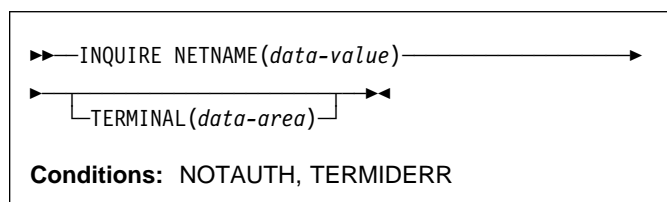
Performance data is recorded separately for each LUW.

TIME(*cvda*)

returns a CVDA value identifying whether the performance class time-stamp fields returned to an application using the COLLECT STATISTICS MONITOR command will be expressed in local or Greenwich mean time. The value of this option has no effect on the other classes of monitoring data. See the *CICS/ESA Customization Guide* for information on the SMF header. CVDA values are:

INQUIRE NETNAME

Retrieve information about a VTAM terminal or session.



Description

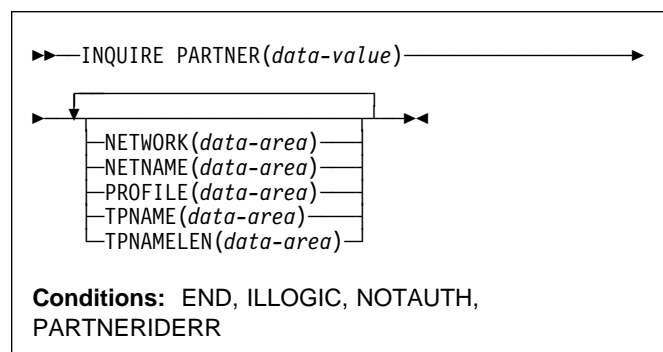
The INQUIRE NETNAME command returns information about a particular terminal or session, just as the INQUIRE TERMINAL command does. The primary difference is that you identify the terminal by its VTAM network identifier, instead of its CICS terminal identifier. Consequently, the roles of the NETNAME and TERMINAL options are reversed; NETNAME is required, and you supply a data-value containing the 8-character network identifier of the terminal about which you are inquiring. TERMINAL is optional. If you use it, CICS returns the corresponding 4-character CICS terminal identifier in the data-area you provide.

The other options for INQUIRE TERMINAL return the same information in an INQUIRE NETNAME command as they do in an INQUIRE TERMINAL command. However, you cannot browse using INQUIRE NETNAME; you must use INQUIRE TERMINAL if you use the START, NEXT, or END options. In addition, you can use INQUIRE NETNAME only for VTAM terminals and sessions; non-VTAM terminals and MRO sessions cannot be the subject of INQUIRE NETNAME commands.

See “INQUIRE TERMINAL” on page 114 for the options and conditions that apply to the INQUIRE NETNAME command.

INQUIRE PARTNER

Retrieve information about a partner.



Description

The INQUIRE PARTNER command returns information about a partner from the partner resource table.

Browsing

You can also browse through all of the partners defined in your system by using the browse options (START, NEXT, and END) on INQUIRE PARTNER commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

NETNAME(data-area)

returns the 8-character name of the VTAM node in which the partner is located.

NETWORK(data-area)

returns the 8-character name of the network in which the partner is located. If this value is blank, the partner is in the same network as your CICS system.

PARTNER(data-value)

specifies the 8-character name of the partner about which you are inquiring. This is the name assigned in its PARTNER resource definition.

PROFILE(data-area)

returns the 8-character name of the PROFILE definition specified in the PARTNER definition.

TPNAME(data-area)

returns the name of the remote transaction program that runs on the partner LU (from the TPNAME or XTPNAME value in the PARTNER resource definition). This name may be up to 64 characters long; you can determine the actual length with the TPNAMELEN option.

INQUIRE PROFILE

TPNAMELEN(*data-area*)

returns a halfword binary field giving the length in bytes of the information returned in TPNAME.

Conditions

END

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

PARTNERIDERR

RESP2 values:

- 1 The partner cannot be found.
- 2 Partner Resource Manager (PRM) is not active, because it failed to initialize during CICS initialization.

INQUIRE PROFILE

Determine whether a transaction profile is installed.

▶▶—INQUIRE PROFILE(*data-value*)—◀◀

Conditions: END, ILLOGIC, NOTAUTH, PROFILEIDERR

Description

The INQUIRE PROFILE command allows you to determine whether a particular PROFILE definition is installed in your CICS system. The command has no options; you get a normal response if the profile about which you inquire is installed in your CICS system, and a PROFILEIDERR exception condition if it is not.

Browsing

You can also use the INQUIRE PROFILE command in browse form (the START, NEXT, and END options) to obtain the names of all of the profiles installed in your system. See "Browsing resource definitions" on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

PROFILE(*data-value*)

specifies the 8-character name of the profile about which you are inquiring.

Conditions

END

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

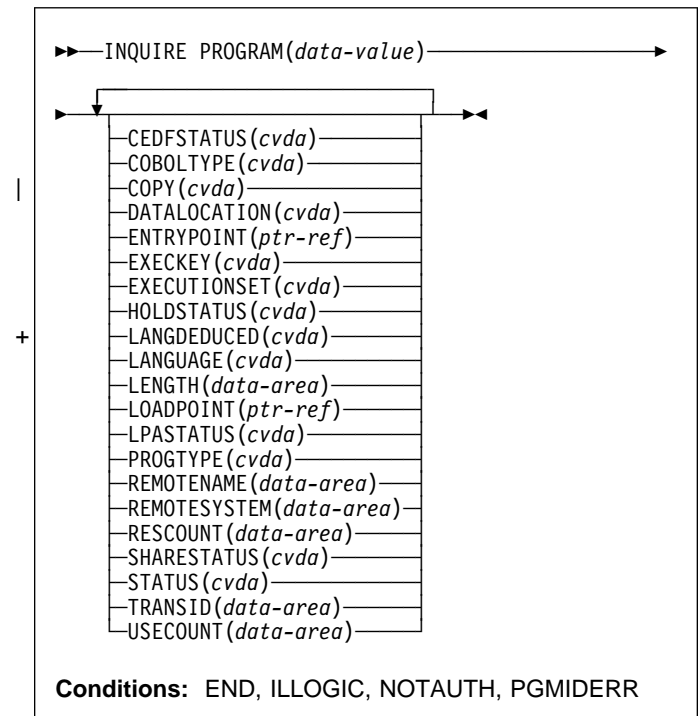
PROFILEIDERR

RESP2 values:

- 1 The profile cannot be found.

INQUIRE PROGRAM

Retrieve information about a program, map set, or partition set.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

The INQUIRE PROGRAM command returns information about a particular program, map set, or partition set installed in your CICS system. All of these resources are load modules and, therefore, CICS uses the same INQUIRE command for all three. To avoid confusion, we use the word **module** to refer to the object of your inquiry, except in some cases where the option applies only to executable programs.

CICS determines the information you request from both the resource definition and, where applicable, the load module. Information from the module takes precedence over that in the definition if there is a conflict. However, CICS will inspect a module only if it is already loaded and is the copy currently available for use. CICS will not do a load for an INQUIRE PROGRAM command, nor attempt to autoinstall a resource for which it has no definition.

INQUIRE PROGRAM

Browsing

You can also browse through the definitions of these three types of resources in your system by using the browse options (START, AT, NEXT, and END) on INQUIRE PROGRAM commands. In browse mode, the definitions are returned in alphabetical order, and you can specify a starting point with the AT option if you wish. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

CEDFSTATUS(*cvda*) (programs only)

returns a CVDA value indicating the action taken by the execution diagnostic facility (EDF) transaction if this module is executed under EDF. CVDA values are:

CEDF

EDF diagnostic screens are displayed. If the program was translated with the EDF option, all EDF screens are displayed; if it was translated with NOEDF, only the program initiation and termination screens appear.

NOCEDF

No EDF screens are displayed.

NOTAPPLIC

EDF is not applicable because the module is a remote program, a map set, or a partition set.

COBOLTYPE(*cvda*) (programs only)

returns a CVDA value indicating the type of COBOL in which the module is written, if it is a COBOL program. The type is determined by inspecting the load module. CVDA values are:

COBOL

The module is an OS/VS COBOL program.

COBOLII

The module is a VS COBOL II program.

NOTAPPLIC

COBOL type does not apply, because the module is a remote program, a map set, or a partition set.

NOTINIT

The module is defined as a COBOL program, but the type cannot be determined because the module has not been loaded yet.

COPY(*cvda*)

returns a CVDA value indicating whether a new copy of the module is required to make it available for use. This requirement occurs after CICS attempts to load the module and cannot find it, because CICS marks it “not loadable” to avoid the overhead of further load attempts. To make the module available again, you need to issue a SET PROGRAM COPY command or its CEMT equivalent. You should ensure that the program exists in one of the libraries in the DFHRPL concatenation before doing so. CVDA values are:

NOTREQUIRED

A new copy is not required.

REQUIRED

A new copy is required.

DATALOCATION(*cvda*) (programs only)

returns a CVDA value indicating whether this module can accept data addresses higher than 16MB. CVDA values are:

ANY The program can accept an address above 16MB.

BELOW

The program requires any data address returned to it from CICS to be less than 16MB.

NOTAPPLIC

The option is not applicable because the module is a remote program, a map set, or a partition set.

ENTRYPOINT(*ptr-ref*)

returns the entry point of the module, if it is loaded. The top bit of the address is set on if the addressing mode is 31 and off if it is 24. If the module has not been loaded or is a remote program, a null pointer (X'FF000000') is returned.

EXECKEY(*cvda*) (programs only)

returns a CVDA value indicating the storage key of the module, if it is an executable program. The storage key can limit the areas of storage that the program can access, depending on other variables. (See the ISOLATEST option of the INQUIRE TASK and INQUIRE TRANSACTION commands, the STOREPROTECT and TRANISOLATE options of the INQUIRE SYSTEM command, and the general discussion of storage protection in the *CICS/ESA System Definition Guide*. CVDA values are:

CICSEXECKEY

The program executes in CICS key.

NOTAPPLIC

The module is a remote program, a map set, or a partition set.

USEREXECKEY

The program executes in user key.

EXECUTIONSET(*cvda*) (programs only)

returns a CVDA value indicating whether the module is restricted to the distributed program link subset of the CICS API. EXECUTIONSET applies only to executable programs, and governs the API only when a program is invoked locally. (When it is invoked remotely—that is, executing at or below the level of a program invoked by a distributed program link—a program is always restricted to this subset.) CVDA values are:

DPLSUBSET

The program is always restricted.

FULLAPI

The program is not restricted unless invoked remotely.

NOTAPPLIC

EXECUTIONSET does not apply because the module is a remote program, a map set, or a partition set.

HOLDSTATUS(*cvda*)

returns a CVDA value indicating whether a copy of the module is currently loaded with the HOLD option. CVDA values are:

HOLD

A copy is currently loaded with the HOLD option.

NOHOLD

No copy is currently loaded with the HOLD option.

NOTAPPLIC

The module is not currently loaded, or is a remote program.

+ **LANGDEDUCED(*cvda*) (programs only)**

returns a CVDA value indicating the language (that is, the module is loaded), or the defined language from the resource definition if not. CVDA values are:

+ ASSEMBLER

The language is assembler.

+ C

The language is C/370.

+ COBOL

The language is VS COBOL II or COBOL/370.

+ LE370

The module, whatever its language, was compiled to run in the LE/370 environment.

+ NOTAPPLIC

LANGUAGE does not apply because the module is a remote program, a map set, or a partition set.

+ NOTDEFINED

The language was not specified in the resource definition.

+ PLI or PL1

The language is PL/I.

LANGUAGE(*cvda*) (programs only)

returns a CVDA value indicating the language that is defined on the program resource definition. CVDA values are:

ASSEMBLER

The language is assembler.

C

The language is C/370.

COBOL

The language is VS COBOL II or COBOL/370.

LE370

The module, whatever its language, was compiled to run in the LE/370 environment.

NOTAPPLIC

LANGUAGE does not apply because the module is a remote program, a map set, or a partition set.

| NOTDEFINED

+ The language was not specified in the resource definition.

| PLI or PL1

The language is PL/I.

LENGTH(*data-area*)

returns a fullword binary field giving the length of the module in bytes. A value of 0 is returned if the module has not been loaded in the current CICS session. A value of -1 is returned if it is a remote program.

LOADPOINT(*ptr-ref*)

returns the load address of the module. If it is not currently loaded, a null pointer (X'FF000000') is returned.

LPASTATUS(*cvda*)

| returns a CVDA value indicating whether the module resided in the link pack area when it was last used. CVDA values are:

| LPA The copy used was in the link pack area (LPA) or the extended link pack area (ELPA).

| NOTAPPLIC

| The module has not been used or is a remote program.

| NOTLPA

| The copy used was in CICS dynamic storage.

PROGRAM(*data-value*)

specifies the 8-character name of the program, map set, or partition set about which you are inquiring.

PROGTYPE(*cvda*)

returns a CVDA value indicating the type of module. CVDA values are:

| MAPSET

| The module is a map set. (MAP is still a synonym for MAPSET, but MAPSET is the preferred CVDA value.)

| PARTITIONSET

| The module is a partition set.

| PROGRAM

| The module is an executable program.

REMOTENAME(*data-area*) (programs only)

returns the 8-character name by which the module is known in the CICS region named in the REMOTESYSTEM option of its PROGRAM definition. REMOTENAME applies only to programs, and only to those defined to be remote; for local programs, map sets, and partitions sets, the value returned is blanks.

REMOTESYSTEM(*data-area*) (programs only)

returns the 4-character name of the CICS region in which the module is defined (from the

INQUIRE PROGRAM

REMOTESYSTEM value in the PROGRAM definition). It applies only to programs, and only to those defined to be remote; for local programs, map sets, and partition sets, the value returned is blanks.

RESCOUNT(*data-area*)

returns a fullword binary field giving the number of separate uses of this module that are taking place at the time of this inquiry. A value of -1 is returned if the module is a remote program.

SHARESTATUS(*cvda*)

returns a CVDA value indicating where CICS should obtain the module the next time a new copy is required. CVDA values are:

NOTAPPLIC

SHARESTATUS is not applicable because the module is a remote program.

PRIVATE

The module will be loaded from the concatenated libraries named on the DFHRPL DD statement.

SHARED

The LPA copy will be used, if one is available. If it is not, the module will be loaded as if SHARESTATUS were PRIVATE.

STATUS(*cvda*)

returns a CVDA value indicating whether the module is available for use. CVDA values are:

DISABLED

The module is not available for use.

ENABLED

The module is available for use.

TRANSID(*data-area*) (programs only)

returns the 4-character name of the transaction under which this module, which must be a program, executes remotely (that is, the transaction identifier the remote region assigns to the task created there to execute it when a task in the local region LINKs to it). This value comes from the TRANSID option value in the PROGRAM definition and applies only to programs defined as remote; for local programs, map sets, and partition sets, and when no TRANSID is specified for a remote program, the value returned is blanks.

USECOUNT(*data-area*)

returns a fullword binary field giving the total number of times the module has been used since the start of the current CICS session. A value of -1 is returned if the program is remote.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.
- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

PGMIDERR

RESP2 values:

- 1 The program cannot be found. If this error occurs on an INQUIRE PROGRAM NEXT, an earlier cataloging error has made a PROGRAM, MAPSET or PARTITIONSET definition unusable, and the definition must be discarded and reinstalled.

Conditions

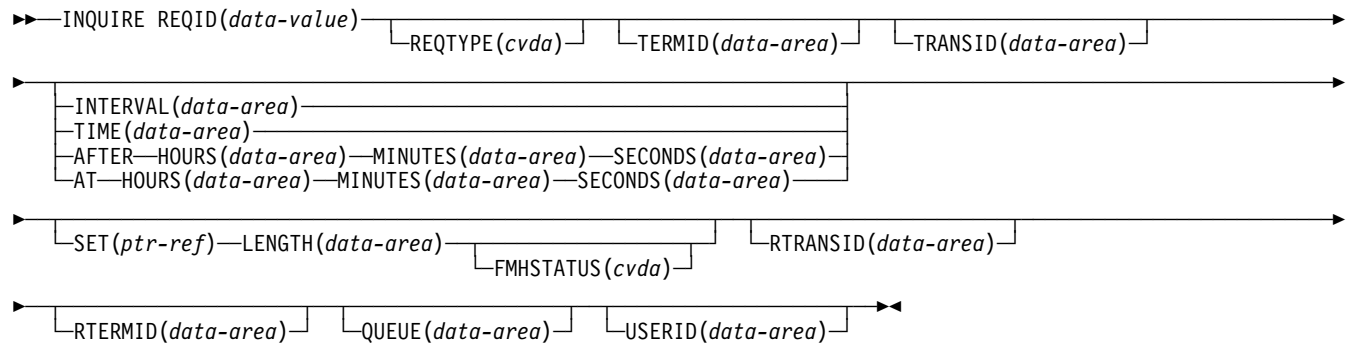
END

RESP2 values:

- 2 There are no more resource definitions of this type.

INQUIRE REQID

Retrieve information about a queued request.



Conditions: INVREQ, NOTAUTH, NOTFND

For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

- Separate hours, minutes and seconds, which you obtain by specifying HOURS, MINUTES, and SECONDS with either AT or AFTER.

Description

The INQUIRE REQID command returns information about a queued request. A queued request results from a DELAY, POST, ROUTE, or START command with a nonzero expiry time, and it lasts until that time. For a DELAY command, expiry time is the end of the delay; for a POST, it is the time at which posting is to occur; for a ROUTE, it is the time at which the message is to be delivered; and for a START, it is the time at which CICS is to create the requested task.

After a request expires, you cannot inquire about it with INQUIRE REQID, even if the action requested is not complete. (A request to START a transaction may be delayed beyond expiry time, for example, waiting for the terminal it requires.)

Requests are identified by the REQID value in the originating command (or assigned by CICS, if omitted in the command). REQID values should be and normally are unique; however, if there is more than one queued request with the same identifier, INQUIRE REQID returns information about the one that will expire soonest.

Expiry time can be expressed either as an interval (the length of time from your INQUIRE to expiry) or as an absolute value (the length of time after the midnight previous to your INQUIRE). If expiry is before midnight of the current day, absolute time is the same as time-of-day, using a 24-hour clock. You can request either form, regardless of how the time was specified in the command that created the request.

There are also two formats for expiry time, whether it is an absolute value or an interval:

- A 4-byte packed decimal composite (0hhmmss+), which you obtain by using the TIME or INTERVAL option.

Expiry time and request type (the type of command that produced it) are available for any queued request. For START requests additional information is available, including data passed from the starting to the started task.

START commands have four options for passing data. The FROM option is primary, and allows you to pass data of variable length, but three others—QUEUE, RTERMID, and RTRANSID—allow you to pass small items of fixed length. They are intended for convenience in conveying resource names to the started transaction, but are not restricted to that purpose. All four data items are kept in temporary storage, and consequently are subject to explicit deletion by another task. If data that you request in an INQUIRE REQID command has been deleted from temporary storage or cannot be read because of an I/O error, CICS raises the INVREQ condition.

Browsing

You also can browse through all of the queued requests by using the browse options (START, NEXT, and END) on INQUIRE REQID commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

INQUIRE REQID

Options

AFTER

requests that CICS return the expiry time (in the HOURS, MINUTES, and SECONDS options) as the **interval** between the current time and the expiry time.

AT requests that CICS return the expiry time (in the HOURS, MINUTES, and SECONDS options) as an **absolute** value (following the midnight preceding this inquiry).

FMHSTATUS(*cvda*)

returns a CVDA value indicating whether the data passed in the FROM option of the command that created this request contains function management headers. FMHSTATUS applies only to requests resulting from ROUTE commands, or START commands that specify FROM. CVDA values are:

FMH The data contains a function management header.

NOFMH

The data does not contain a function management header.

NOTAPPLIC

The request did not result from a ROUTE or START command, or there was no FROM data.

HOURS(*data-area*)

returns a fullword binary field giving the hours portion of the expiry time (see the AT and AFTER options).

INTERVAL(*data-area*)

returns the expiry time as an interval from the current time. The value is a 4-byte packed decimal number in the format 0hhmmss+.

LENGTH(*data-area*)

returns a halfword binary field giving the length of the data passed in the FROM option of the command that created this request. It applies only to requests resulting from ROUTE commands, or START commands that specify FROM; for other requests, the value returned is zero.

MINUTES(*data-area*)

returns a fullword binary field giving the minutes portion of the expiry time (see the AT and AFTER options).

QUEUE(*data-area*)

returns the 8-byte field passed in the QUEUE option of the START command that created this request. It applies only to requests resulting from START commands that specify QUEUE; for other requests, the value returned is blanks.

REQID(*data-value*)

specifies the 8-byte identifier of the request about which you are inquiring. This is the value specified in the REQID option of the command that generated the request (or assigned by CICS if REQID was omitted).

REQTYPE(*cvda*)

returns a CVDA value indicating the type of command that created this request. CVDA values are:

DELAY

A DELAY command created this request.

POST

A POST command created this request.

ROUTE

A ROUTE command created this request.

START

A START command created this request.

RTERMID(*data-area*)

returns the 4-byte field passed in the RTERMID option of the START command that created this request. It applies only to requests resulting from START commands that specify RTERMID; for other requests, the value returned is blanks.

RTRANSID(*data-area*)

returns the 4-byte field passed in the RTRANSID option of the START command that created this request. It applies only to requests resulting from START commands that specify RTRANSID; for other requests, the value returned is blanks.

SECONDS(*data-area*)

returns a fullword binary field giving the seconds portion of the expiry time (see the AT and AFTER options).

SET(*ptr-ref*)

returns the address of the data passed in the FROM option of the command which created this request. It applies only to requests resulting from ROUTE commands, or START commands that specify FROM; for other requests, the value returned is the null pointer (X'FF000000').

TERMID(*data-area*)

returns the 4-character terminal identifier that was specified in the TERMID option of the START command that created the request. It applies only to requests originating from START commands that specify a terminal; for other requests, the value returned is blanks.

TIME(*data-area*)

returns the expiry time as an absolute value measured from the midnight preceding this INQUIRE command. The value is a 4-byte packed decimal number in the format 0hhmmss+.

TRANSID(*data-area*)

returns the 4-character transaction identifier that was specified in the TRANSID option of the command that created the request. It applies only to requests originating from ROUTE or START commands; for other requests, the value returned is blanks.

USERID(*data-area*)

returns the 8-character identifier of the user associated with the task that issued the command that created this

| request. USERID applies only to requests resulting from
| ROUTE or START commands; for other requests, the
| value returned is blanks.

| **Conditions**

| **INVREQ**

| RESP2 values:

- | **3** An I/O error occurred while an attempt was made
| to read data from temporary storage for the SET,
| QUEUE, RTERMID or RTRANSID option.
- | **4** Data required for the SET, QUEUE, RTERMID or
| RTRANSID option cannot be returned because it
| has been deleted from temporary storage.

| **NOTAUTH**

| RESP2 values:

- | **100** The user associated with the issuing task is not
| authorized to use this command.

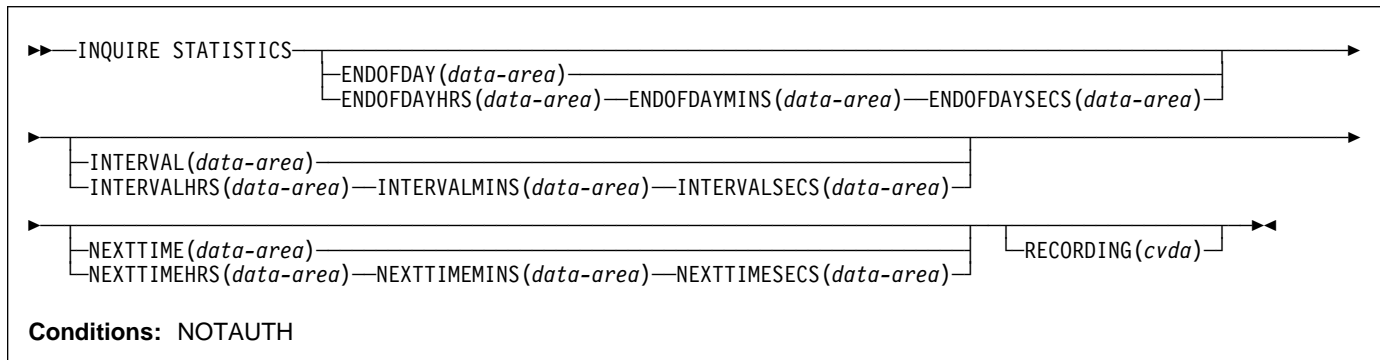
| **NOTFND**

| RESP2 values:

- | **1** The REQID cannot be found.

INQUIRE STATISTICS

Retrieve statistics information.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE STATISTICS command returns information about the recording of CICS resource and system statistics. CICS records system statistics periodically if the RECORDING switch is on, at a frequency governed by the + INTERVAL option. These statistics are called **interval + statistics**. At end-of-day time (the ENDOFDAY option), + CICS records **end-of-day statistics**—which are the statistics + for the interval since the last resetting— whether or not the switch is on, ensuring that statistics are written at least once a day. Recording occurs on a system management facility (SMF) data set, and the counts are reset after recording.

| There are two formats for each of the time values that you
| can retrieve with this command (the end-of-day time, the
| recording interval, and the next time that recording will
| occur):

- | • A 4-byte packed decimal composite (0hhmmss+), which
| you obtain by using the ENDOFDAY, INTERVAL and
| NEXTTIME options.
- | • Separate hours, minutes and seconds, which you obtain
| by specifying the ENDOFDAYHRS, ENDOFDAYMINS,
| and ENDOFDAYSECS options (instead of ENDOFDAY),
| INTERVALHRS, INTERVALMINS, and INTERVALSECS
| (instead of INTERVAL) and NEXTTIMEHRS,
| NEXTTIMEMINS, and NEXTTIMESECS (instead of
| NEXTTIME).

The *CICS/ESA Performance Guide* contains more detail
+ about CICS statistics, and the SET STATISTICS command
+ on page 172 describes the relationship between the interval
and end-of-day times.

Options

ENDOFDAY(*data-area*)

returns the end-of-day time, as a 4-byte packed decimal field in the format 0hhmmss+. End-of-day time is expressed in local time.

ENDOFDAYHRS(*data-area*)

returns the hours component of the end-of-day time, in fullword binary form.

ENDOFDAYMINS(*data-area*)

returns the minutes component of the end-of-day time, in fullword binary form.

ENDOFDAYSECS(*data-area*)

returns the seconds component of the end-of-day time, in fullword binary form.

INTERVAL(*data-area*)

returns a 4-byte packed decimal field giving the recording interval for system statistics.

INTERVALHRS(*data-area*)

returns the hours component of the recording interval, in fullword binary form.

INTERVALMINS(*data-area*)

returns the minutes component of the recording interval, in fullword binary form.

INTERVALSECS(*data-area*)

returns the seconds component of the recording interval, in fullword binary form.

NEXTTIME(*data-area*)

returns a 4-byte packed decimal field giving the time at which statistics will be recorded next (assuming that the RECORDING switch is not changed from its current value). This is the end-of-day time if RECORDING is currently off, and the earlier of end-of-day and the end of the current interval otherwise.

NEXTTIMEHRS(*data-area*)

returns the hours component of the next recording time, in fullword binary form.

NEXTTIMEMINS(*data-area*)

returns the minutes component of the next recording time, in fullword binary format.

NEXTTIMESECS(*data-area*)

returns the seconds component of the next recording time, in fullword binary format.

RECORDING(*cvda*)

controls the recording of interval statistics. End-of-day
statistics, requested statistics and unsolicited statistics
are always recorded, irrespective of the setting of the
RECORDING option." (Unsolicited statistics are
resource statistics, recorded when the resource is
discarded. Requested statistics are those called for by a
PERFORM STATISTICS RECORD command, described
on page 144, or by a CEMT PERFORM STATISTICS
transaction.)

CVDA values for the RECORDING option are:

OFF switches off the recording of interval statistics.

ON switches on the recording of interval statistics.

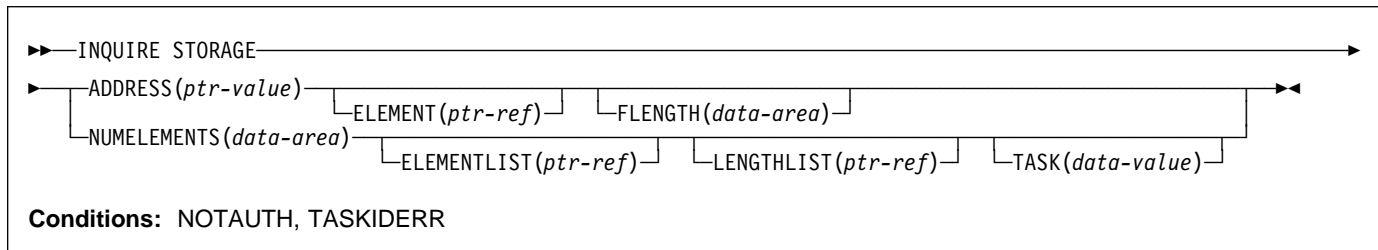
Conditions**NOTAUTH**

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

INQUIRE STORAGE

Retrieve information about task storage.



Description

The INQUIRE STORAGE command has two functions. You can use it to get a list of the task storage areas associated with a particular task (using the NUMELEMENTS option), or you can use it to find the length and starting address of a particular area of storage (using the ADDRESS option). INQUIRE STORAGE applies only to storage allocated to user tasks, which are tasks executing user-defined transactions or the CICS-supplied transactions normally invoked by an operator.

Note: INQUIRE STORAGE does not return information about areas obtained through an EXEC CICS GETMAIN if the command specifies the SHARED option.

Options

ADDRESS(ptr-value)
specifies that you are inquiring about a single area of storage and identifies the area. The address you specify can be anywhere within the area about which you are inquiring; it does not have to be the start of it. CICS returns the length of the area (in FLENGTH) and its starting address (in ELEMENT) if it is a valid element of user task storage.

ELEMENT(ptr-ref)
returns the starting address of the storage area containing the address provided in the ADDRESS option, if the area is user task storage. This is the first byte of the area available for task data, not the preceding storage management control information, if any. If the area is not user task storage, the address returned is nulls.

ELEMENTLIST(ptr-ref)
returns the address of a list of the addresses of all areas of task storage for the task specified in the TASK option. Each address points to the first byte available for data storage, not to preceding storage management control information, if any. The number of addresses in this list is the NUMELEMENTS option value. (Addresses are 4 bytes long, and therefore the length of the list in bytes is 4 times NUMELEMENTS.)

CICS obtains the storage for this list and frees it when the inquiring task ends, or issues another INQUIRE STORAGE command with ELEMENTLIST or LENGTHLIST, or issues an INQUIRE TASK LIST; the task cannot free the storage itself.

FLENGTH(data-area)
returns a fullword binary field giving the length of the storage area containing the address provided in the ADDRESS option. This is the length of the part available for task data; it does not include storage management control information at the beginning or end of the area, if any. If the area is not user task storage, the length returned is -1.

LENGTHLIST(ptr-ref)
returns the address of a list of fullword binary lengths. Each entry in this list is the length of the storage area to which the corresponding entry in the ELEMENTLIST list points. These lengths are the amounts available for data storage and do not include storage management control information, if any.

CICS obtains the storage for this list and frees it when the inquiring task ends, or issues another INQUIRE STORAGE command with ELEMENTLIST or LENGTHLIST, or issues an INQUIRE TASK LIST; the task cannot free the storage itself.

NUMELEMENTS(data-area)
indicates that you are requesting a list of the task storage areas for the task indicated in the TASK option. CICS returns the number of areas, in fullword binary form, in the data area you provide. If you request an ELEMENTLIST or LENGTHLIST, this value is the number of entries in the list.

TASK(data-value)
specifies, as a 4-byte packed decimal value, the task number for which you are requesting a storage list. If you omit this option but include NUMELEMENTS, CICS assumes the inquiry is for the task issuing the INQUIRE STORAGE command.

Conditions**NOTAUTH**

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

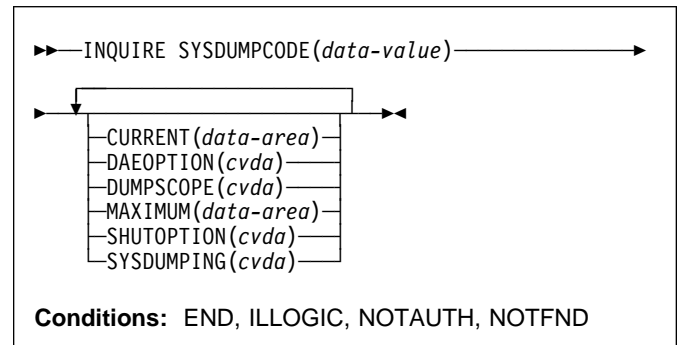
TASKIDERR

RESP2 values:

- 1** The task number does not exist.
2 The task number designates a system task, not a user task.

INQUIRE SYSDUMPCODE

+ Retrieve information about a system dump table entry.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

The INQUIRE SYSDUMPCODE command allows you to look at some of the information in a system dump code table entry.

The table entry tells CICS what actions to take when a system dump request with this code occurs, and how many times to take them (the MAXIMUM option); requests received after the maximum are counted (the CURRENT option), but otherwise ignored.

+ CICS provides a system dump table with entries for some + CICS-defined system dump codes. If it receives a dump + request for a code for which it does not have an entry, it + builds one, using default values. You can add your own + entries with the SET SYSDUMPCODE command or a CEMT + transaction. Entries you add remain over executions of CICS + until a cold start occurs, but the entries that CICS builds are + considered to be temporary and are discarded at shutdown. + Consequently, if you enquire about a code that is not + explicitly defined before it appears in a dump request, you + will get a "not found" response.

Browsing

You can also browse through all of the entries in the system dump code table by using the browse options (START, NEXT and END) on INQUIRE SYSDUMPCODE commands. See "Browsing resource definitions" on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options**CURRENT(data-area)**

returns a fullword binary field giving the number of dump requests with this dump code made since the count was

INQUIRE SYSDUMPCODE

last reset. (The count is reset automatically at CICS shutdown and can be reset explicitly with a SET SYSDUMPCODE command or its CEMT equivalent.) The count includes requests that do not result in a dump because either CICS or MVS suppressed it.

DAEOPTION

returns a CVDA value identifying whether a dump produced for this dump code is eligible for subsequent suppression by the MVS Dump Analysis and Elimination (DAE) component. CVDA values are:

DAE The dump is eligible for DAE suppression.

NODAE

The dump is not eligible for DAE suppression—if CICS determines that a dump should be written, MVS will not suppress it. (However, be aware of the SUPPRESS and SUPPRESSALL options in the ADYSETxx parmlib member. These are controlled by the VRADAE and VRANODAE keys in the SDWA. They may lead to dump suppression even though NODAE is set here. For information about DAE, SUPPRESS, and SUPPRESSALL, see the *MVS ESA Planning: Problem Determination and Recovery* manual.

#

DUMPSCOPE(*cvda*)

returns a CVDA value indicating whether a request for a dump with this dump code should cause an SDUMP (system dump) request to be sent to related MVS images.

A related image is one that contains a CICS region doing work on behalf of your CICS region—specifically, a region that has one or more tasks doing work under the same APPC token as a task in your region.

The sending of SDUMP requests occurs only when the table entry for this code specifies a dump (that is, the SYSDUMPING value is SYSDUMP), and only in a sysplex environment executing under MVS/ESA 5.1 and the MVS workload manager.

CVDA values are:

LOCAL

SDUMP requests are not to be sent.

RELATED

SDUMP requests are to be sent.

MAXIMUM(*data-area*)

returns a fullword binary field giving the maximum number of dumps with this code that CICS will take. A value of 999 means

#

APAR PQ01620

changed this sentence

the default, 'no limit'.

SHUTOPTION(*cvda*)

returns a CVDA value indicating whether the CICS system is to be shut down after a request for a dump with this dump code. CVDA values are:

NOSHUTDOWN

The CICS system is not to be shut down.

SHUTDOWN

The CICS system is to be shut down.

SYSDUMPCODE(*data-value*)

specifies the 8-character system dump code about which you are inquiring. A valid code contains no leading or imbedded blanks.

SYSDUMPING(*cvda*)

returns a CVDA value indicating whether a dump request with this code should produce a dump or not. Even when a dump is specified, CICS will take one only when the CURRENT value is no greater than the MAXIMUM, and when system dumps are not suppressed globally (see the DUMPING option of the SET SYSTEM command on page 177). MVS may also be allowed to suppress the dump if appropriate (the DAE option). CVDA values are:

+
+

NOSYSDDUMP

A dump is not to be taken.

SYSDUMP

A dump is to be taken.

Note: Dumps from the kernel domain of CICS are not subject to suppression and are taken regardless of SYSDUMPCODE value.

Conditions

END

RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

NOTFND

RESP2 values:

1 The named dump code cannot be found.

INQUIRE SYSTEM

Retrieve CICS system information.

INQUIRE SYSTEM	
AKP (data-area)	
CDSASIZE (data-area)	
CICSSSTATUS (cvda)	
CICSSSYS (data-area)	
CMDPROTECT (cvda)	
DFLTUSER (data-area)	
DSALIMIT (data-area)	
DTRPROGRAM (data-area)	
DUMPING (cvda)	
ECDSASIZE (data-area)	
EDSALIMIT (data-area)	
ERDSASIZE (data-area)	
ESDSASIZE (data-area)	
EUDSASIZE (data-area)	
GMMTEXT (data-area) GMMLENGTH (data-area)	
GMMTRANID (data-area)	
INITSTATUS (cvda)	
JOBNAME (data-area)	
MAXTASKS (data-area)	
MROBATCH (data-area)	
OPREL (data-area)	
OPSYS (data-area)	
PROGAUTOCTLG (cvda)	
PROGAUTOEXIT (data-area)	
PROGAUTOINST (cvda)	
PRTYAGING (data-area)	
RDSASIZE (data-area)	
REENTPROTECT (cvda)	
RELEASE (data-area)	
RUNAWAY (data-area)	
SCANDELAY (data-area)	
SDSASIZE (data-area)	
SECURITYMGR (cvda)	
SHUTSTATUS (cvda)	
SOSSSTATUS (cvda)	
STARTUP (cvda)	
STARTUPDATE (data-area)	
STOREPROTECT (cvda)	
TIME (data-area)	
TRANSISOLATE (cvda)	
UDSASIZE (data-area)	
XRFSTATUS (cvda)	

Conditions:
NOTAUTH

For more information about the use of CVDA's, see "CICS-value data areas (CVDA's)" on page 7.

Description

The INQUIRE SYSTEM command returns information about the CICS system under which the task issuing the command is executing.

Many of the options in this command correspond to options in the system initialization table (SIT) and take their initial values from the SIT. Some of these can be changed by a subsequent SET SYSTEM command or its CEMT equivalent.

Other options return information about the CICS or MVS release levels, and still others return information determined solely by the current state of the system. Table 2 on page 100 indicates where the option values come from, and, in the case of those set initially by the SIT, the name of the corresponding option. For these options, the *CICS/ESA System Definition Guide* is a good source of additional information. For state options, the *CICS/ESA Customization Guide* is the primary source.

INQUIRE SYSTEM

Table 2. INQUIRE SYSTEM options

Option	Origin
AKP	AKPFREQ in SIT
CDSASIZE	System state
CICSSTATUS	System state
CICSSYS	System state
CMDPROTECT	CMDPROT in SIT
DFLTUSER	DFLTUSER in SIT
DSALIMIT	DSALIM in SIT
DTRPROGRAM	DTRPGM in SIT
DUMPING	DUMP in SIT
ECDSASIZE	System state
EDSALIMIT	EDSALIM in SIT
ERDSASIZE	System state
ESDSASIZE	System state
EUDSASIZE	System state
GMMTEXT, GMMLNGTH	GMTEXT in SIT
GMMTRANID	GMTRAN in SIT
INITSTATUS	System state
JOBNAME	JCL or catalogued procedure
MAXTASKS	MXT in SIT
MROBATCH	MROBTCH in SIT
OPREL	MVS system code
OPSYS	MVS system code
PROGAUTOCTLG	PGAICTLG in SIT
PROGAUTOEXIT	PGAEXIT in SIT
PROGAUTOINST	PGAIPGM in SIT
PRTYAGING	PRTYAGE in SIT
RDSASIZE	System state
REENPROTECT	RENTPGM in SIT
RELEASE	CICS system code
RUNAWAY	ICVR in SIT
SCANDELAY	ICVTSD in SIT
SDSASIZE	System state
SECURITYMGR	SEC in SIT
SHUTSTATUS	System state
SOSSTATUS	System state
STARTUP	System state
STARTUPDATE	System state
STOREPROTECT	STGPROT in SIT and hardware
TIME	ICV in SIT
TRANISOLATE	TRANISO in SIT and hardware
UDSASIZE	System state
XRFSTATUS	XRF in SIT and system state

Note: The CSCS, ECSCS, ERSCS, EUSCS, and USCS options, each of which returned the size of the storage “cushion” for a particular dynamic storage area, are obsolete in CICS/ESA 4.1. The translator accepts them and gives a warning. At run-time, the data areas provided are left unchanged.

Options

AKP(*data-area*)

returns a fullword binary field giving the activity keypoint trigger value, which is the number of logging operations between the taking of keypoints.

- + A value of zero means that keypoints are not being
- + taken.

CDSASIZE(*data-area*)

returns the current size in bytes of the CICS dynamic storage area (CDSA), in fullword binary form. It includes both storage in use and storage available for use. This size is calculated and managed by CICS automatically, within the overall limit for dynamic storage areas that reside below the 16MB boundary (the DSALIMIT option value).

CICSSTATUS(*cvda*)

returns a CVDA value identifying the current execution status of CICS. CVDA values are:

ACTIVE

CICS is fully active.

FINALQUIESCE

CICS is in the final quiesce stage of shutdown. Programs in the second stage of the program list table for shutdown (PLTSD) are run during this stage.

FIRSTQUIESCE

CICS is in the first quiesce stage of shutdown. Programs in the first stage of the PLTSD are run during this stage.

STARTUP

CICS is starting up but is not yet fully active. Programs in the program list table for program initiation (PLTPI) are run during startup. See the INITSTATUS option for further information.

CICSSYS(*data-area*)

returns a 1-character value identifying the operating system for which the running CICS system has been built. A value of “X” represents MVS/ESA.

CMDPROTECT(*cvda*)

returns a CVDA value indicating whether command protection is active or not. With command protection active, when a task issues a command, CICS verifies that the task has write access to the first byte of every area into which CICS is to return information. If any area fails the test, an AEYD abend occurs.

The CVDA values are:

CMDPROT

Command protection is active.

NOCMDPROT

Command protection is not active.

DFLTUSER(*data-area*)

returns the 8-character identifier of the default user for this CICS region.

DSALIMIT(*data-area*)

returns a fullword binary field giving the maximum amount of storage, in bytes, within which CICS can dynamically allocate storage for the four individual dynamic storage areas that reside below the 16MB boundary. (See the CDSASIZE, RDSASIZE, SDSASIZE and UDSASIZE options of this command.)

DTRPROGRAM(*data-area*)

returns the 8-character name of the program controlling the dynamic routing of transactions.

DUMPING(*cvda*)

returns a CVDA value indicating whether the taking of CICS system dumps is suppressed. CVDA values are:

NOSYSDUMP

System dumps are suppressed.

SYSDUMP

System dumps are not suppressed.

ECDSASIZE(*data-area*)

returns the current size in bytes of the extended CICS dynamic storage area (ECDSA), in fullword binary form. It includes both storage in use and storage available for use. This size is calculated and managed by CICS automatically, within the overall limit for dynamic storage areas that reside above the 16MB boundary (the EDSALIMIT option value).

EDSALIMIT(*data-area*)

returns a fullword binary field giving the maximum amount of storage, in bytes, within which CICS can dynamically allocate storage for the four individual dynamic storage areas that reside above the 16MB boundary. (See the ECDSASIZE, ERDSASIZE, ESDSASIZE and EUDSASIZE options of this command.)

ERDSASIZE(*data-area*)

returns the current size in bytes of the extended read-only dynamic storage area (ERDSA), in fullword binary form. It includes both storage in use and storage available for use. This size is calculated and managed by CICS automatically, within the overall limit for dynamic storage areas that reside above the 16MB boundary (the EDSALIMIT option value).

ESDSASIZE(*data-area*)

returns the current size in bytes of the extended shared dynamic storage area (ESDSA), in fullword binary form. It includes both storage in use and storage available for use. This size is calculated and managed by CICS automatically, within the overall limit for dynamic storage areas that reside above the 16MB boundary (the EDSALIMIT option value).

EUDSASIZE(*data-area*)

returns the current size in bytes of the extended user dynamic storage area (EUDSA), in fullword binary form. It includes both storage in use and storage available for use. This size is calculated and managed by CICS automatically, within the overall limit for dynamic storage areas that reside above the 16MB boundary (the EDSALIMIT option value).

GMMLength(*data-area*)

returns a halfword binary field giving the length in bytes of the “good morning” message text.

GMMTEXT(*data-area*)

returns the “good morning” message text in the *data-area* you provide, which must long enough to accomodate it. The maximum length of any “good morning” message is 246 bytes; the actual length is returned in the GMMLENGTH option value.

GMMTRANID(*data-area*)

returns the 4-character name of the transaction that generates the “good morning” message.

INITSTATUS(*cvda*)

returns a fullword binary field giving the initialization status of the CICS system. CVDA values are:

FIRSTINIT

First stage of CICS initialization.

INITCOMPLETE

CICS initialization is complete.

SECONDINIT

Second stage of initialization.

THIRDINIT

Third stage of initialization.

See the *CICS/ESA Customization Guide* for more information about CICS initialization.

JOBNAME(*data-area*)

returns the 8-character MVS jobname under which CICS is running.

MAXTASKS(*data-area*)

returns a fullword binary field giving the maximum number of tasks that can be eligible for dispatch at any one time in this CICS system. Both active and suspended tasks count toward this limit, but tasks that have not reached the point of initial dispatch do not. System tasks such as terminal and journal control tasks do not count in CICS/ESA 4.1 either, although they did in earlier releases.

MROBATCH(*data-area*)

returns a fullword binary field giving the number of events that must occur, from a list of MRO and DASD I/O events on which CICS is waiting, before CICS is posted explicitly to process them.

OPREL(*data-area*)

returns a halfword binary field giving the release number of the operating system under which CICS is currently running. The value returned is 10 times the formal release number (the version number is not represented). For example, MVS/ESA Version 3 Release 2.1 would produce a value of 21.

OPSYS(*data-area*)

returns a 1-character value identifying the operating system under which CICS is running. A value of “X” represents MVS/ESA.

INQUIRE SYSTEM

PROGAUTOCTLG(*cvda*)

+ returns a CVDA value indicating whether and when autoinstalled program definitions are cataloged. Cataloged definitions are restored on a warm or emergency restart. Those not catalogued are discarded at shutdown, and must be installed again if they are used in a subsequent execution of CICS.

+ Decisions to catalog are made both at initial installation and whenever an autoinstalled definition is modified, and are based on the PROGAUTOCTLG value at the time. CVDA values are:

CTLGALL

+ Definitions are cataloged both when installed and when modified.

CTLGMODIFY

+ Definitions are cataloged only when modified.

CTLGNONE

+ Definitions are not cataloged.

PROGAUTOEXIT(*data-area*)

| returns the 8-character name of the user-provided program that is called by the CICS program autoinstall code to provide a model definition.

PROGAUTOINST(*cvda*)

| returns a CVDA value indicating whether autoinstall for programs is active or inactive. When a task requests a program, map set or partition set that is not defined, CICS attempts to create a definition for it automatically if autoinstall for programs is active. If not, CICS raises the PGMIDERR exceptional condition. CVDA values are:

AUTOACTIVE

| Autoinstall for programs is active.

AUTOINACTIVE

| Autoinstall for programs is not active.

PRTYAGING(*data-area*)

| returns a fullword binary field giving the rate at which CICS increases the priority of a task waiting for dispatch. CICS increases the task priority by 1 after each PRTYAGING milliseconds of wait time without a dispatch.

RDSASIZE(*data-area*)

| returns the current size in bytes of the read-only dynamic storage area (RDSA), in fullword binary form. It includes both storage in use and storage available for use. This size is calculated and managed by CICS automatically, within the overall limit for dynamic storage areas that reside below the 16MB boundary (the DSALIMIT option value).

REENTPROTECT(*cvda*)

| returns a CVDA value indicating whether storage for reentrant programs (the RDSA and ERDSA) is in key 0 or CICS key. MVS key 0 storage is write protected from programs running in CICS key or user key; programs in CICS key storage are protected only from those running in user key when CICS key and user key are different

(that is, when storage protection is active). CVDA values are:

REENTPROT

Read-only DSAs are in key 0 storage.

NOREENTPROT

Read-only DSAs are in CICS-key storage.

RELEASE(*data-area*)

| returns a 4-character string giving the version and release numbers of the CICS code running. For example, "0330" represents CICS/ESA Version 3 Release 3, and "0410" represents CICS/ESA Version 4 Release 1.

RUNAWAY(*data-area*)

| returns a fullword binary field giving the default value for runaway task time. This value is used for any task executing a transaction whose profile does not specify runaway task time (see the RUNAWAY option of the INQUIRE TRANSACTION command on page 132).

SCANDELAY(*data-area*)

| returns a fullword binary field giving the maximum number of milliseconds between a user task making a terminal I/O request and CICS dispatching the terminal control task to process it. This value is sometimes called the "terminal scan delay," and is set by the ICVTSD option in the system initialization table.

SDSASIZE(*data-area*)

| returns the current size in bytes of the shared dynamic storage area (SDSA), in fullword binary form. It includes both storage in use and storage available for use. This size is calculated and managed by CICS automatically, within the overall limit for dynamic storage areas that reside below the 16MB boundary (the DSALIMIT option value).

SECURITYMGR(*cvda*)

| returns a CVDA value identifying whether an external security manager (such as RACF) is active in the system, or whether no security is being used. CVDA values are:

EXTSECURITY

An external security manager is active.

NOSECURITY

No security is being used.

SHUTSTATUS(*cvda*)

| returns a CVDA value indicating the shutdown status of CICS (see the CICSSTATUS option on page 100). CVDA values are:

CANCELLED

CICS is canceled.

CONTROLSHUT

CICS is performing a controlled shutdown (that is, a normal shutdown with a warm keypoint).

| SHUTDOWN
| CICS is performing an immediate shutdown.

| NOTAPPLIC
| CICS is not shutting down.

| **SOSSTATUS(*cvda*)**
| returns a CVDA value indicating whether CICS is short
| on storage. CVDA values are:

| NOTSOS
| CICS is not short on storage in any of the dynamic
| storage areas.

| SOS CICS is short on storage in at least one dynamic
| storage area above and at least one below the
| 16MB line.

| SOSABOVE
| CICS is short on storage in at least one dynamic
| storage area above 16MB, but none below.

| SOSBELOW
| CICS is short on storage in at least one dynamic
| storage area below 16MB, but none above.

| **STARTUP(*cvda*)**
| returns a CVDA value indicating how the current
| execution of CICS started. CVDA values are:

| COLDSTART
| CICS performed a cold start (either on request or
| because of the global catalog had been initialized).

| EMERGENCY
| CICS performed an emergency restart because
| the previous run did not shut down normally.

| WARMSTART
| CICS performed a warm restart following the
| normal shutdown of the previous run.

| **STARTUPDATE(*data-area*)**
| returns a 4-byte packed-decimal field containing the date
| on which the current execution of CICS started. The
+ date is in the form **0cyyddd+**, where **c.** is the century
+ code (**0** for the 1900s, **1** for 2000–2099), and **yy** is the
+ low-order two digits of the year and **ddd** is the
+ sequential day of the year.

| **STOREPROTECT(*cvda*)**
| returns a CVDA value indicating whether storage
| protection is active or not. For storage protection to be
| active, it must be requested (the STGPROT option in the
| system initialization table), and it must be supported by
| the hardware. CVDA values are:

ACTIVE
Storage protection is active.

INACTIVE
Storage protection is not active.

| **TIME(*data-area*)**
| returns a fullword binary field giving the maximum
| interval in milliseconds for which CICS gives control to
| the operating system if no tasks are ready for dispatch.

This value is set by the ICV option in the system
initialization table and is sometimes called the “region
exit time interval.”

| **TRANISOLATE(*cvda*)**
| returns a CVDA value indicating whether transaction
| isolation is active or not. For it to be active, both
| transaction isolation and storage protection must be
| requested (the TRANISO and STGPROT options in the
| SIT), and it must be supported by the hardware. CVDA
| values are:

ACTIVE
Transaction isolation is active.

INACTIVE
Transaction isolation is not active.

| **UDSASIZE(*data-area*)**
| returns the current size in bytes of the user dynamic
| storage area (UDSA), in fullword binary form. It includes
| both storage in use and storage available for use. This
| size is calculated and managed by CICS automatically,
| within the overall limit for dynamic storage areas that
| reside below the 16MB boundary (the DSALIMIT option
| value).

| **XRFSTATUS(*cvda*)**
| returns a CVDA value indicating whether the current
| execution of CICS started as an active or alternate
| region under the extended recovery facility (XRF).
| CVDA values are:

NOTAPPLIC
CICS is running without XRF support (XRF=NO in
the system initialization table).

PRIMARY
CICS started as the active region.

TAKEOVER
CICS started as the alternate region.

Conditions

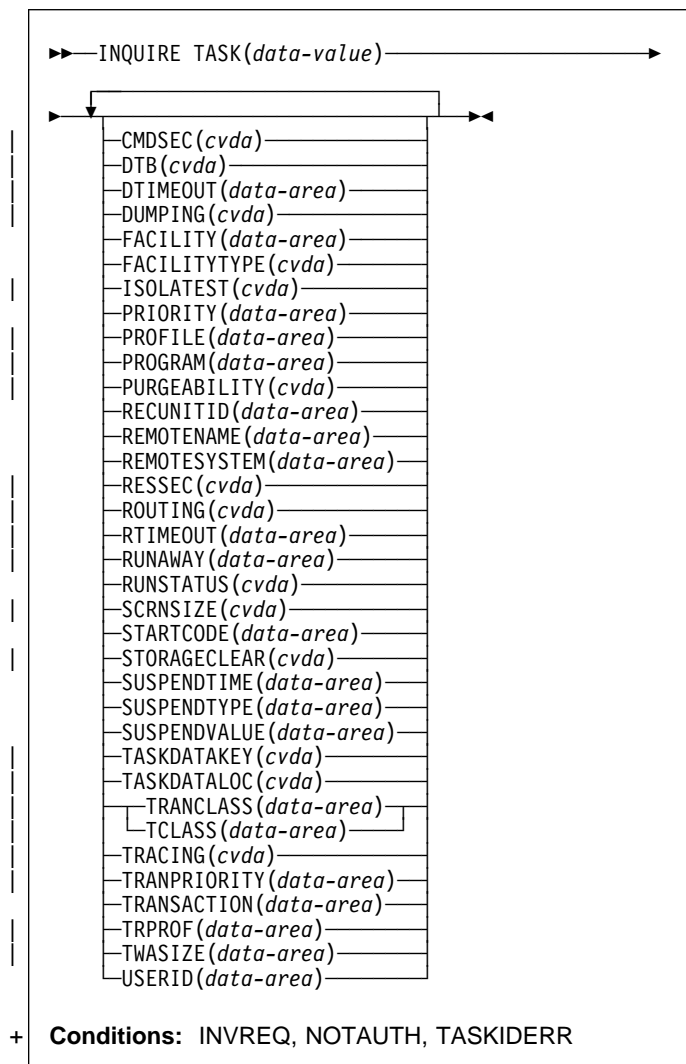
NOTAUTH
RESP2 values:

100 The user associated with the issuing task is not
authorized to use this command.

INQUIRE TASK

INQUIRE TASK

Retrieve information about a user task.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

For more information about device properties, see the # TYPETERM chapter in the *CICS/ESA Resource Definition # Guide*.

Description

The INQUIRE TASK command returns information about a specific user task. User tasks are those associated with user-defined transactions or with CICS-supplied transactions that are normally invoked by an operator.

Many of the options available on this command are the same as those available on the INQUIRE TRANSACTION command, because a task obtains most of its characteristics from the definition of the transaction it is executing.

However, these properties are determined at task initiation. If the transaction definition is changed after the task begins, the task may have a different value for a property than the current transaction definition. Task values can also be changed with a SET TASK command or its CEMT equivalent.

In addition, the INQUIRE TASK command always produces information about the task you specify on the **local** CICS system. You need to keep this in mind for tasks that are subject to routing or that issue LINK commands that may be shipped to another system.

Whenever a task is executed wholly or in part on a system other than the one on which it originates, there is a matching task on the remote system. The task on the originating system takes its characteristics from the definition *on that system* of the transaction it is to execute. The corresponding task on the remote system (if routing takes place or the task issues distributed program links) takes its characteristics from the definition of whatever transaction *on the remote system* that the originating system tells the remote system to use. This remote transaction may have different properties from those of the transaction on the originating system. (It may or may not have a different name; in the case of static transaction routing, the name of the transaction in the remote system comes from the REMOTENAME option of the transaction in the local system.)

Consequently, an inquiry about the task on the originating system may produce entirely different results from an inquiry about the corresponding task on the remote system. For the same reason, a task that issues distributed program links may get a different result from an INQUIRE TASK about itself (taking the task number from the EIB) in a program executing remotely than from the same command in a program executing locally.

Options

- | **CMDSEC**(cvda)
 - + returns a CVDA value indicating whether the definition of
 - + the TRANSACTION the task is executing specifies
 - | command security. CVDA values are:
 - | CMDSECNO
 - | Command security is not specified.
 - | CMDSECYES
 - | Command security is specified.
 - + When a task being checked issues a system
 - | programming command, CICS calls the external security
 - | manager (ESM) to verify that the user associated with
 - | the task has authority to use these commands.
 - + A task is command-checked only when an ESM is active
 - + and either the CMDSEC value for the task is
 - + CMDSECYES or the system initialization option
 - + CMDSEC value is ALWAYS (see the SECURITYMGR
 - + option of the INQUIRE SYSTEM command on page 102

+ and the *CICS/ESA Resource Definition Guide* for more
+ information).

DTB(*cvda*)

returns a CVDA value indicating how uncommitted changes made to recoverable resources by this task are handled if the task fails. CVDA values are:

BACKOUT

Uncommitted changes are backed out.

COMMIT

Uncommitted changes are committed.

WAIT

The disposition of uncommitted changes is deferred until further information is available; locks on the changed resources are held until the disposition is resolved.

The DTB value is set by the INDOUBT option of the definition of the TRANSACTION this task is executing.

DTIMEOUT(*data-area*)

returns a fullword binary field giving the deadlock time-out interval, in seconds. CICS abends a task that waits longer than its deadlock timeout value for a locked resource.

DUMPING(*cvda*)

returns a CVDA value indicating whether CICS should take a transaction dump if the task terminates abnormally. CVDA values are:

NOTRANDUMP

No dump is taken.

TRANDUMP

A dump is taken.

This value applies only to abend dumps and has no effect on DUMP TRANSACTION commands.

FACILITY(*data-area*)

returns the 4-character name of the facility associated with initiation of this task, if that facility is a transient data queue or a terminal or system. If the task was initiated otherwise, the facility value is blanks. The FACILITYTYPE option tells you what type of facility caused task initiation, and therefore what FACILITY represents.

FACILITYTYPE(*cvda*)

returns a CVDA value identifying the type of facility that initiated this task. CVDA values are:

DEST

CICS initiated the task to process a transient data queue which had reached trigger level; the FACILITY option returns the name of queue.

TASK

Another task initiated the task with a START command that did not specify a terminal, or CICS created the task internally; the FACILITY option returns blanks in this case.

TERM

Either the task was initiated to process unsolicited input or another task initiated the task with a START command with the TERMID option. In the first case the FACILITY option returns the name of the terminal that sent the input, and in the second, it returns the terminal named in TERMID.

ISOLATEST(*cvda*)

returns a CVDA value indicating whether the task is defined as isolated or not. Isolation limits the access, for both read and write, of user-key programs to task storage. A program executing in user key on behalf of an isolated task can access the task storage of only that task, and this storage cannot be accessed by programs executing in user key on behalf of other tasks. Isolation does not affect access by CICS-key programs and does not apply to storage with the SHARED attribute or any other nontask storage.

The value of ISOLATEST is taken from the definition of the TRANSACTION the task is executing when the task is created. For a task defined as isolated to execute isolated, transaction isolation for the system must also be active (see the TRANISOLATE option of the INQUIRE SYSTEM command on page 103). CVDA values are:

ISOLATE

The task is defined as isolated.

NOISOLATE

The task is defined as not isolated.

PRIORITY(*data-area*)

returns a fullword binary field giving the total priority of the task. Total priority is the sum of the priority of the user associated with the task, the priority of the terminal which is the principal facility, and the priority of the TRANSACTION being executed (see the TRANPRIORITY option).

PROFILE(*data-area*)

returns the 8-character name of the PROFILE for the TRANSACTION this task is executing.

PROGRAM(*data-area*)

returns the 8-character name of the program executed first in this task.

PURGEABILITY(*cvda*)

returns a CVDA value indicating whether CICS is allowed to purge this task (that is, to terminate it abnormally). Purge requests come from SET TASK PURGE commands (or CEMT equivalents), and CICS can generate them internally to reclaim resources to relieve a system stall condition. CVDA values are:

NOTPURGEABLE

The task cannot be purged.

PURGEABLE

The task can be purged.

INQUIRE TASK

| The PURGEABILITY value is set initially by the
| SPURGE option in the definition of the TRANSACTION
| this task is executing.

RECUNITID(*data-area*)

returns the 8-byte identifier of the current unit of recovery in this task. CICS assigns this value at the start of the logical unit of work (unit of recovery), using the system clock value at that time to ensure a unique value. Unit-of-recovery values are used to synchronize recovery operations among CICS and other resource managers, such as IMS and DB2.

REMOTENAME(*data-area*)

returns the 4-character name assigned in the REMOTENAME option of the definition of the TRANSACTION which this task is executing. When CICS routes a task statically, REMOTENAME is the name of the transaction that the partner task on the remote system executes. Consequently REMOTENAME is significant to the task about which you are inquiring only if it is subject to routing.

CICS returns blanks if the TRANSACTION definition does not specify REMOTENAME.

REMOTESYSTEM(*data-area*)

returns the 4-character name assigned in the REMOTESYSTEM option of the definition of the TRANSACTION which this task is executing. When CICS routes a task statically, REMOTESYSTEM is the name of the CONNECTION definition of the system to which the task is routed. Like REMOTENAME, REMOTESYSTEM is significant to the task about which you are inquiring only if it is subject to routing.

CICS returns blanks if the TRANSACTION definition does not specify REMOTESYSTEM.

RESSEC(*cvda*)

+ returns a CVDA value indicating whether the definition of
+ the TRANSACTION the task is executing specifies
| resource-level security checking. CVDA values are:

RESSECNO

+ Command security is not specified.

RESSECYES

+ Command security is specified.

+ When a task is being checked, CICS verifies on each
| command that the user associated with the task has
| authority to access the resource named in the way
| requested.

+ A task is checked only when an external security
+ manager is active and either the RESSEC value for the
+ task is RESSECYES or the system initialization option
+ RESSEC value is ALWAYS (see the SECURITYMGR
+ option of the INQUIRE SYSTEM command on page 102,
+ and the *CICS/ESA Resource Definition Guide* for more
+ information).

ROUTING(*cvda*)

returns a CVDA value indicating whether the transaction this task is executing specifies dynamic routing or not (in the DYNAMIC option in the TRANSACTION definition). Dynamic routing occurs just before the initial dispatch of a task, and therefore this value indicates whether dynamic routing may have occurred (if the task is already in execution) or may yet occur (if it has not yet been dispatched). CVDA values are:

DYNAMIC

Dynamic routing applies.

STATIC

Dynamic routing does not apply.

RTIMEOUT(*data-area*)

returns a fullword binary field giving the read time-out interval, in seconds. CICS abends a task if it waits for input longer than its read time-out value. The RTIMEOUT value is set by the RTIMEOUT option in the PROFILE definition associated with the TRANSACTION this task is executing.

RUNAWAY(*data-area*)

returns the "runaway task" time for this task, in milliseconds, as a fullword binary value. If a task keeps control of the processor for more than this interval on a single dispatch, CICS assumes it is in a loop and abends it. If the value is zero, CICS does not monitor the task for a runaway condition.

RUNSTATUS(*cvda*)

returns a CVDA value indicating the dispatch status of the task. CVDA values are:

DISPATCHABLE

The task is ready to run.

RUNNING

The task is running.

SUSPENDED

The task is not ready to run.

SCRNSIZE(*cvda*)

returns a CVDA value indicating whether the alternate or the default screen size applies to this task. CVDA values are:

ALTERNATE

Alternate screen size applies.

DEFAULT

Default screen size applies.

The SCRNSIZE value is set by the same-named option in the PROFILE definition associated with the TRANSACTION this task is executing.

STARTCODE(*data-area*)

returns a 2-character value indicating how this task started. Possible values are:

- D The task was initiated to process a distributed programming link (DPL) command that did not specify the SYNCONRETURN option. (The task is not allowed to issue syncpoints.)
- DS The task was initiated to process a distributed programming link (DPL) command containing the SYNCONRETURN option. (The task is allowed to issue syncpoints.)
- QD CICS initiated the task to process a transient data queue that had reached trigger level.
- S Another task initiated this one, using a START command that did not pass data in the FROM option.
- SD Another task initiated this one, using a START command that passed data in the FROM option.
- SZ The task was initiated with a FEPI START command (see the *CICS/ESA Front End Programming Interface User's Guide* for further information).
- TO The task was initiated to process unsolicited input from a terminal (or another system), and the transaction to be executed was determined from the input.
- TP The task was initiated to process unsolicited input or in response to a RETURN IMMEDIATE command in another task. In either case, the transaction to be executed was preset (in the RETURN command or in the associated TERMINAL definition) without reference to input.
- U CICS created the task internally.

STORAGECLEAR(*cvda*)

returns a CVDA value indicating whether CICS should clear storage that is released from this task (to prevent other tasks accidentally viewing confidential data).
CVDA values are:

CLEAR

Storage will be cleared.

NOCLEAR

Storage will not be cleared.

SUSPENDTIME(*data-area*)

returns a fullword binary field giving the number of seconds (rounded down) for which the task has been suspended since last dispatch, if its RUNSTATUS value is SUSPENDED. If the task is running or dispatchable, the SUSPENDTIME value is 0.

SUSPENDTYPE(*data-area*)

returns an 8-character text string indicating why this task is suspended, if it is (blanks are returned for tasks that are running or dispatchable). See the SUSPENDVALUE option also.

SUSPENDVALUE(*data-area*)

returns the 8-character name of the resource for which this task is waiting (the name of the file if the task is

enqueued on a record, for example). SUSPENDVALUE applies only to suspended tasks; if the task is running or dispatchable, the value returned is blanks.

For information on the values that can appear in the SUSPENDTYPE and SUSPENDVALUE options, and how they can be used as an aid in problem determination, see the “resource type” and “resource name” details in the *CICS/ESA Problem Determination Guide*.

TASK(*data-value*)

specifies the 4-byte packed-decimal sequence number of the task about which you are inquiring.

TASKDATAKEY(*cvda*)

returns a CVDA value indicating the storage key in which CICS obtains storage for this task. This includes the task life-time storage—the transaction work area (TWA) and the EXEC interface block (EIB)—and the storage that CICS obtains on behalf of programs that run under this task.

See the description of the TASKDATAKEY option in a TRANSACTION resource definition in the *CICS/ESA Resource Definition Guide* for more information.

CVDA values are:

CICSDATAKEY

CICS obtains storage from CICS-key storage.

USERDATAKEY

CICS obtains storage from user-key storage.

The value returned for TASKDATAKEY is taken from the definition of the TRANSACTION that the task is executing. To determine whether storage protection is active (that is, whether user-key has a different value from CICS-key), you need to issue an INQUIRE SYSTEM command with the STOREPROTECT option.

TASKDATALOC(*cvda*)

returns a CVDA value indicating whether task-lifetime storage for this task (CICS control blocks for the task such as the EIB and TWA) should be acquired above or below the 16MB line. CVDA values are:

ANY Task-lifetime storage can be either below or above the 16MB line.

BELOW

Task-lifetime storage must be below the 16MB line.

TCLASS(*data-area*)

returns a fullword binary field giving the number of the transaction class to which this task belongs, if it belongs to a numbered transaction class. This option is retained for compatibility with earlier releases, where transaction classes were numbered from 0 to 10. If the task does not belong to such a class, the value returned is zero. (See the TRANCLASS option.)

INQUIRE TASK

| **TRACING**(*cvda*)

| returns a CVDA value indicating the type of tracing in effect for this task. CVDA values are:

| SPECTRACE

| Tracing for this task is special.

| SPRSTRACE

| Tracing for this task is suppressed.

| STANTRACE

| Tracing for this task is standard.

| For further information on the types of tracing, see the *CICS/ESA Problem Determination Guide* and the description of the CETR transaction in the *CICS/ESA CICS-Supplied Transactions* manual.

| **TRANCLASS**(*data-area*)

| returns the 8-character name of the transaction class to which the task belongs. If the task is not assigned to any class, the default class DFHTCL00 is returned. If the task belongs to a numbered class, the value returned is DFHTCLnn, where **nn** is the two-digit class number.

| **TRANPRIORITY**(*data-area*)

| returns a fullword binary field giving the component of the total priority of the task that came from the PRIORITY option in the definition of the TRANSACTION being executed. (See the PRIORITY option of this command also.)

TRANSACTION(*data-area*)

returns the 4-character name of the transaction that this task is executing.

| **TRPROF**(*data-area*)

| returns the 8-character name of the PROFILE definition used for intersystem flows if the task is routed on an ISC link.

| **TWASIZE**(*data-area*)

| returns a fullword binary field giving the size in bytes of the transaction work area (TWA) for this task.

USERID(*data-area*)

returns the 8-character identifier of the user associated with the task.

Conditions

+ **INVREQ**

+ RESP2 values:

- + **3** TCLASS cannot be returned because the task is in a transaction class that does not correspond to TCLASS values 1 through 10.
- + **10** Neither RTIMEOUT nor SCRNSIZE can be returned because the selected task's PROFILE entry cannot be located.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

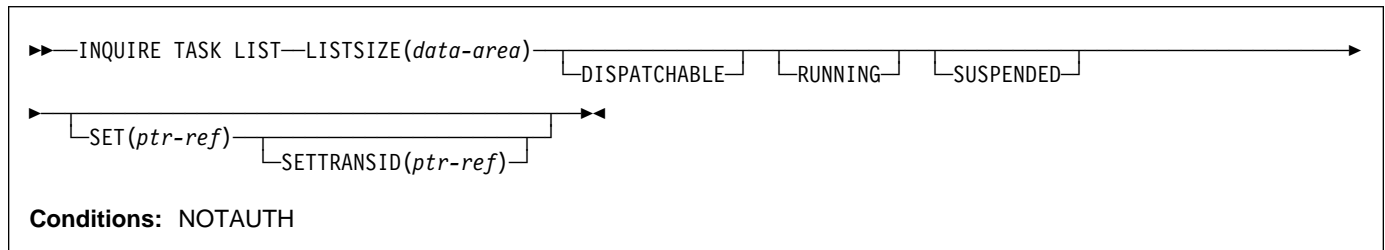
TASKIDERR

RESP2 values:

- 1** The task cannot be found.
- 2** The task is executing a type of transaction which is not subject to this command.

INQUIRE TASK LIST

Retrieve a list of user tasks.



Description

The INQUIRE TASK LIST command returns a list of user tasks. User tasks are those associated with user-defined transactions or with CICS-supplied transactions that are normally invoked by an operator. You can restrict the list to tasks that are DISPATCHABLE (ready to run), RUNNING, or SUSPENDED at the time of the inquiry, or any combination of these.

- + If the program issuing the EXEC CICS INQUIRE TASK LIST command is being run under CEDF, the CEDF task will not appear in the task list because CEDF is pseudoconversational.

Options

DISPATCHABLE

specifies that tasks ready to run (dispatchable) should be included in the task list. These tasks are also included if you specify none of the category options (DISPATCHABLE, RUNNING, and SUSPENDED).

LISTSIZE(*data-area*)

returns a fullword binary field giving the number of tasks in the categories you included in your inquiry. This is the number of entries in the lists that the SET and SETTRANSID options produce. If there are no tasks in the categories requested, LISTSIZE contains zero.

RUNNING

specifies that the task executing (the one issuing the command) should be included in the task list. It is also included if you specify none of the category options (DISPATCHABLE, RUNNING, and SUSPENDED).

SET(*ptr-ref*)

returns the address of a list of 4-byte packed-decimal task numbers. Each entry in the list identifies a task in one of the categories requested (see the DISPATCHABLE, RUNNING and REQUESTED options). If there are no tasks in the categories requested, the SET pointer contains a null value.

CICS obtains the storage for this list and frees it when the inquiring task issues another INQUIRE TASK LIST or ends; the task cannot free the storage itself.

SETTRANSID(*ptr-ref*)

returns the address of a list of 4-byte transaction identifiers. Each entry in the list is the name of the transaction that the task in the corresponding entry in the SET list is executing. If there are no tasks in the categories that you have specified, the SETTRANSID pointer contains a null value.

CICS obtains the storage for this list and frees it when the inquiring task issues another INQUIRE TASK LIST or ends; the task cannot free the storage itself.

SUSPENDED

specifies that suspended tasks (tasks waiting for some event or condition) should be included in the task list. For this purpose, tasks which have not reached the point of initial dispatch, either because the task class to which they belong is at its maximum or because the maximum for the system has been reached, are considered suspended. Suspended tasks are also included if you specify none of the category options (DISPATCHABLE, RUNNING, and SUSPENDED).

Conditions

NOTAUTH

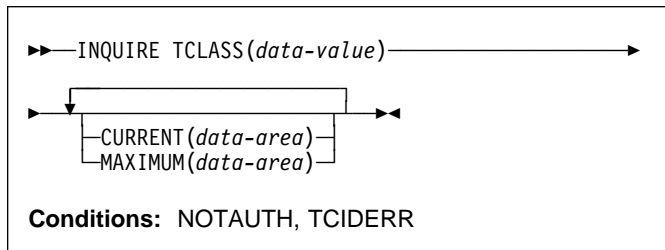
RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

INQUIRE TCLASS

INQUIRE TCLASS

Retrieve information about a transaction class.



Description

The INQUIRE TCLASS command allows you to determine the current and maximum numbers of tasks within an installation-defined transaction class. This command is limited to the numbered classes of earlier releases of CICS and is retained for compatibility with those releases. The INQUIRE TRANCLASS command, described on page 127, has the same function and can be used for either the old numbered or the new named classes.

Options

CURRENT(*data-area*)

returns a fullword binary field giving the current number of tasks in the class about which you are inquiring. This number includes both tasks that are running and tasks that have not yet been dispatched because the maximum for either the class or the system has been reached. (See the MAXIMUM option of this command and the MAXTASKS option of the INQUIRE SYSTEM command for more about these limits.) The CURRENT value corresponds to the sum of the ACTIVE and QUEUED values in an INQUIRE TRANCLASS command, and therefore can exceed the MAXIMUM value.

MAXIMUM(*data-area*)

returns a fullword binary field giving the largest number of tasks that are allowed to run concurrently in the class about which you are inquiring. (This value corresponds to the MAXACTIVE value in an INQUIRE TRANCLASS command.)

TCLASS(*data-value*)

specifies the number of the task class about which you are inquiring, in fullword binary form. The number must be in the range 0–10.

Conditions

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

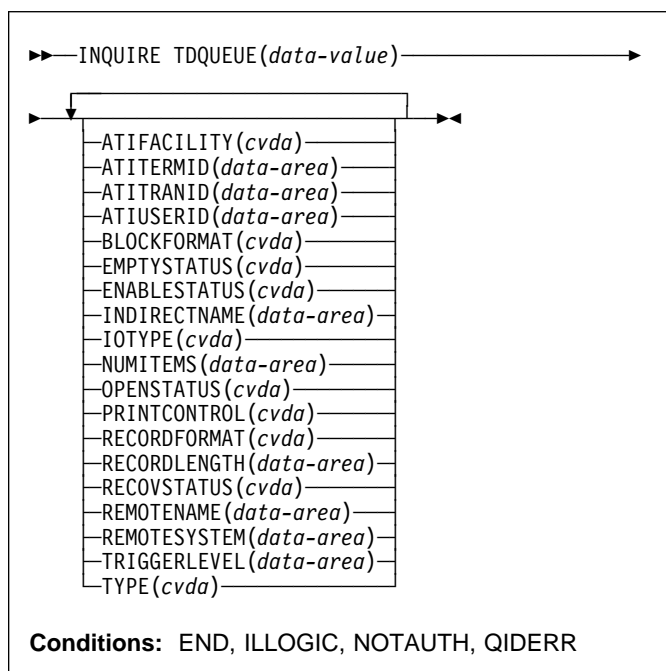
TCIDERR

RESP2 values:

1 The named task class cannot be found.

INQUIRE TDQUEUE

Retrieve information about a transient data queue.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE TDQUEUE command retrieves information about a particular transient data queue.

Transient data queues, also called destinations, are defined in the destination control table (DCT) of CICS. There are two basic types: **intrapartition** and **extrapartition**. Intrapartition queues are managed and stored entirely by CICS, and are subject to automatic task initiation (ATI). ATI means that when the number of items on the queue reaches the value in the TRIGGERLEVEL option, CICS automatically creates a task to process the queue.

An extrapartition queue is an MVS sequential data set (or a spool file). Extrapartition queues are not subject to ATI, and consequently the associated options produce null values. Furthermore, if the data set is not open, CICS may not be able to determine some of the values, such as BLOCKFORMAT and RECORDFORMAT. Null values, explained in “Command format” on page 4, are returned in such cases.

Two other queue types appear in the DCT: **indirect** and **remote**, both of which point, eventually, to one of the basic types.

An indirect queue points to another queue on the same CICS system, and is essentially an alias for the other queue.

When you name an indirect queue in an INQUIRE TDQUEUE command, CICS returns only the TYPE value (which is INDIRECT) and the name of the queue to which the indirect definition points (the INDIRECTNAME value). You need a second INQUIRE TDQUEUE against the INDIRECTNAME value to determine the characteristics of the underlying queue.

A remote queue is one defined on another CICS system. When you inquire about such a queue, the local CICS system returns only the information it maintains locally about the queue: the TYPE (REMOTE), the system on which it is defined (the REMOTESYSTEM value), its name there (REMOTENAME), and whether it is available to applications on the local system (its ENABLESTATUS).

Browsing

You can also browse through all of the transient data queues defined in your system by using the browse options (START, NEXT, and END) on INQUIRE TDQUEUE commands. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

ATIFACILITY(*cvda*) (intrapartition queues only)

returns a CVDA value indicating whether the queue has a terminal (or session) associated with it. If it does, and CICS creates a task to process the queue because its trigger level has been reached, the terminal is assigned as the principal facility of the task. See also the ATITERMID and ATITRANID options. CVDA values are:

NOTAPPLIC

The queue is not intrapartition.

NOTERMINAL

No terminal is associated with the queue.

TERMINAL

A terminal is associated with the queue.

ATITERMID(*data-area*) (intrapartition queues only)

returns the 4-character name of the terminal or session associated with the queue, if any (see the ATIFACILITY option). Otherwise, blanks are returned.

ATITRANID(*data-area*) (intrapartition queues only)

returns the 4-character identifier of the transaction to be executed when CICS initiates a task automatically to process the queue. This option applies only to intrapartition queues intended for ATI; for other types of queues, and for intrapartition queues where no transaction has been specified in the queue definition, the value returned is blanks.

ATIUSERID(*data-area*) (intrapartition queues only)

returns the 8-byte user identifier associated with the queue. CICS assigns this value to a task it creates to process the queue if there is no terminal associated with

INQUIRE TDQUEUE

the queue. If the queue is not intrapartition, or no transaction is defined for it (the ATITRANID option), blanks are returned.

BLOCKFORMAT(*cvda*) (extrapartition queues only)

returns a CVDA value indicating whether the data set associated with the queue is in blocked record format or not. It applies only to extrapartition queues. CVDA values are:

BLOCKED

The records are blocked.

NOTAPPLIC

The data set is not open, or the queue is not extrapartition.

UNBLOCKED

The records are not blocked.

EMPTYSTATUS(*cvda*) (extrapartition queues only)

returns a CVDA value indicating the state of the queue with regard to space. CICS detects a FULL condition only when a task attempts to add a record and there is no space, and detects EMPTY only when a task attempts to read and there are no records. Consequently, a value of NOTEMPTY is returned **unless** one of these conditions has been detected. EMPTYSTATUS applies only to extrapartition queues. CVDA values are:

EMPTY

The queue is empty.

FULL

The queue is full.

NOTAPPLIC

The option does not apply because the queue is not open or is not extrapartition.

NOTEMPTY

No operation against the queue has indicated that it is either empty or full.

ENABLESTATUS(*cvda*) (all except indirect queues)

returns a CVDA value indicating whether the queue can be accessed by applications. For remote queues, this value reflects whether the local CICS will forward commands to access the queue to the remote system or reject them with a DISABLED exception condition; it does not necessarily reflect the state of the queue on the remote system. CVDA values are:

DISABLED

The queue cannot be accessed by applications. (For extrapartition queues, this value does not necessarily mean that the associated data set is closed.)

ENABLED

The queue can be accessed by applications.

NOTAPPLIC

The queue is indirect.

INDIRECTNAME(*data-area*) (indirect queues only)

returns the 4-character name of the queue that this indirect queue points to. This option applies only to queues defined as indirect; for other types of queues, blanks are returned.

IOTYPE(*cvda*) (extrapartition queues only)

returns a CVDA value indicating whether the queue was defined for INPUT, OUTPUT, or READBACK. CVDA values are:

INPUT

The queue is defined for input and will be read forward.

NOTAPPLIC

The queue is not open or is not an extrapartition queue.

OUTPUT

The queue is defined for output.

READBACK

The queue is defined for input and will be read backward.

NUMITEMS(*data-area*) (intrapartition queues only)

returns a fullword binary field giving the number of items in the queue. A value of -1 is returned if the queue is not intrapartition.

OPENSTATUS(*cvda*) (extrapartition queues only)

returns a CVDA value indicating whether the queue is open, closed, or in an intermediate state. CVDA values are:

CLOSED

The queue is closed.

CLOSING

The queue is closing.

NOTAPPLIC

The queue is not extrapartition.

OPEN

The queue is open.

OPENING

The queue is opening.

PRINTCONTROL(*cvda*) (extrapartition queues only)

returns a CVDA value indicating the type of print control, if any, defined for the queue. Printer control characters appear in the first position of the every record when used. However, CICS does not check this character when records are written to the queue, or remove the character when records are read from the queue; use and enforcement of the printer control conventions are up to the applications using the queue. CVDA values are:

ASACTL

ASA control characters are used.

MCHCTL

Machine control characters are used.

NOCTL

No print control characters are used.

NOTAPPLIC

The queue is not open or is not extrapartition.

RECORDFORMAT(*cvda*) (extrapartition queues only)

returns a CVDA value indicating whether the queue has fixed- or variable-length records. CVDA values are:

FIXED

The queue has fixed-length records.

NOTAPPLIC

The queue is not open or is not extrapartition.

VARIABLE

The queue has variable-length records.

RECORDLENGTH(*data-area*) (extrapartition queues only)

returns a fullword binary field giving the record length (in bytes) for queues having fixed-length records, and the maximum record length for queues having variable-length records. RECORDLENGTH applies only to extrapartition queues; for others, -1 is returned.

RECOVSTATUS(*cvda*) (intrapartition queues only)

returns a CVDA value indicating the type of recovery defined for the queue. Recovery is available only for intrapartition queues. CVDA values are:

LOGICAL

The queue is logically recoverable.

NOTAPPLIC

The queue is not intrapartition.

NOTRECOVABLE

The queue is not recoverable.

PHYSICAL

The queue is physically recoverable.

REMOTENAME(*data-area*) (remote queues only)

returns the 4-character name of this queue in the remote CICS region in which the queue is defined (from the RMTNAME option in its definition). REMOTENAME applies only to queues defined as remote; for other queues the value returned is blanks.

REMOTESYSTEM(*data-area*) (remote queues only)

returns the 4-character name of the CICS region in which the queue is defined (from the SYSIDNT value in its definition). REMOTESYSTEM applies only to queues defined as remote; for other queues the value returned is blanks.

TDQUEUE(*data-value*)

specifies the 4-character name of the transient data queue about which you are inquiring.

TRIGGERLEVEL(*data-area*) (intrapartition only)

returns a fullword binary field giving the number of items the queue must reach before automatic transaction initiation (ATI) occurs. When the queue reaches this depth, CICS invokes a task to process it automatically.

A value of zero means the queue is not subject to ATI; a value of -1 is returned if the queue is not intrapartition.

TYPE(*cvda*)

returns a CVDA value identifying the type of queue. CVDA values are:

EXTRA

The queue is extrapartition.

INDIRECT

The queue is indirect.

INTRA

The queue is intrapartition.

REMOTE

The queue is remote.

Conditions**END**

RESP2 values:

2

There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

1

You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

100

The user associated with the issuing task is not authorized to use this command.

101

The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

QIDERR

RESP2 values:

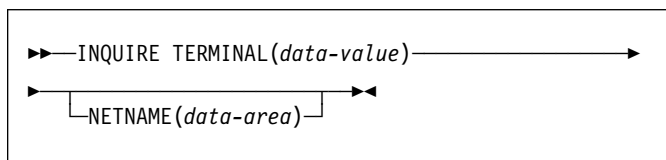
1

The named queue cannot be found.

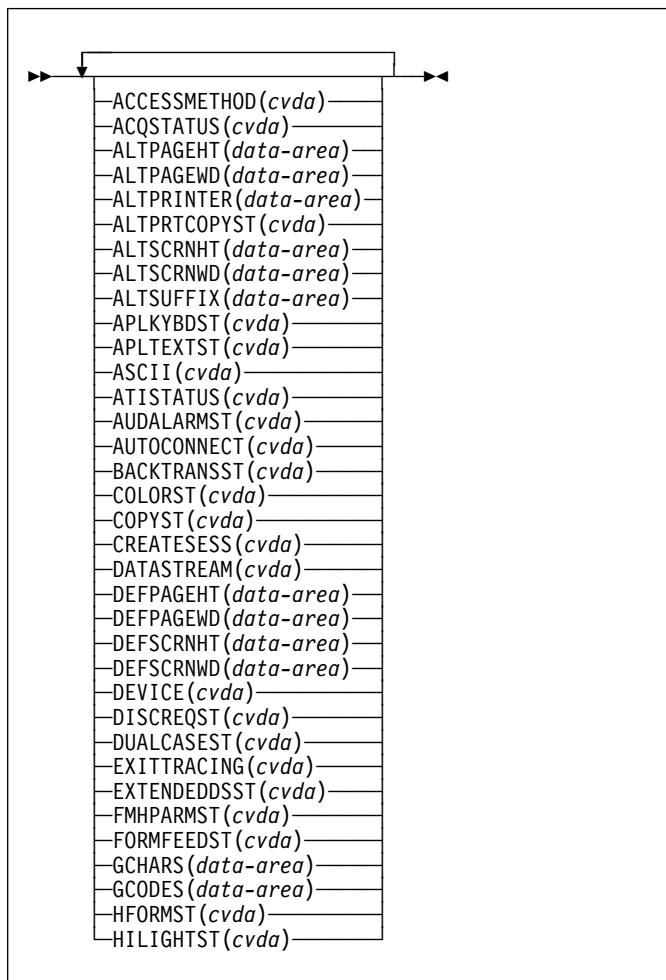
INQUIRE TERMINAL

INQUIRE TERMINAL

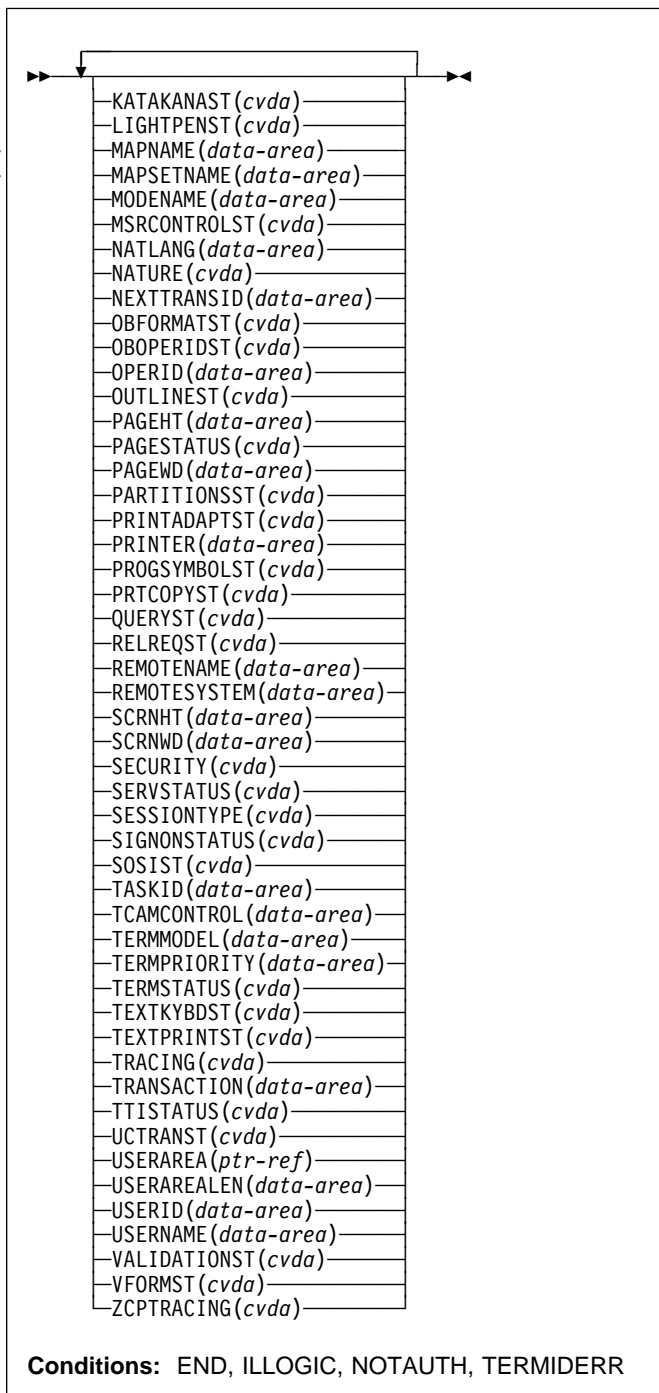
Retrieve information about a terminal or session.



The following options apply to both the INQUIRE TERMINAL and the INQUIRE NETNAME command.



+
+



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

For more information about device properties, see the # TYPETERM chapter in the *CICS/ESA Resource Definition # Guide*.

Description

The INQUIRE TERMINAL and INQUIRE NETNAME commands both return information about a particular terminal or session installed in your CICS system.

You can use them to inquire about any type of terminal resource, including physical terminals owned locally (by the region in which the INQUIRE was issued), remote terminals (terminals defined locally as owned by another region), surrogate terminals (partial definitions which represent terminals owned by another region, shipped to the local region the first time the definition is needed), and models (definitions used only to autoinstall other terminals).

You can also use INQUIRE TERMINAL to inquire about an APPC, LUTYPE6.1 or MRO session or, where there are parallel sessions, session group. To get full detail about the associated connection, however, you must use an INQUIRE CONNECTION command.

Some of the options in this command return system status information, such as whether the terminal is acquired or not, whether it is in use by a task, and so on. Most options, however, reflect the definition of the terminal or session, modified, possibly, by subsequent SET TERMINAL commands or the information obtained from the hardware in a QUERY.

A terminal is specified by a TERMINAL resource definition and the TYPETERM definition to which it points. Characteristics shared by many terminals, such as screen size and 3270 features, are defined by TYPETERM, and those specific to one terminal, such as the name of the associated printer, are in the TERMINAL definition, which may have been autoinstalled. For a session, the CONNECTION defines shared properties and SESSIONS defines specifics. See the *CICS/ESA Resource Definition Guide* for more information about TERMINAL, TYPETERM, SESSIONS, and CONNECTION resource definitions.

In most cases, options of this type have the same name—or one recognizably similar—as the option in the resource definition. Where this is not the case, the option descriptions that follow indicate the corresponding resource options.

INQUIRE NETNAME returns the same information as INQUIRE TERMINAL. With INQUIRE TERMINAL, you identify the object of your inquiry by providing its CICS terminal identifier in the TERMINAL option. NETNAME is optional. If you include it, CICS returns the VTAM network identifier in the data area you provide.

In an INQUIRE NETNAME command, the roles of TERMINAL and NETNAME are reversed. You identify the terminal about which you are inquiring by supplying its network identifier in NETNAME, and CICS returns the corresponding CICS terminal identifier in TERMINAL if you also include that option. TERMINAL must appear before NETNAME (if present) in an INQUIRE TERMINAL command, and vice-versa in an INQUIRE NETNAME command.

All of the other options apply to both commands and return the same information. Not all options apply to all types of terminals, however. In particular, when CICS ships a terminal definition from the owning region to a remote region, an inquiry issued in the owning region (where the definition is of a real terminal) produces more information than an inquiry issued in the remote region, where the definition is a **surrogate** for the one in the owning region.

Browsing

You can also browse through the definitions of all the terminals installed in your system by using the browse options (START, NEXT and END) on INQUIRE TERMINAL commands (the browse options are not available on INQUIRE NETNAME). See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

ACCESSMETHOD(*cvda*)

returns a CVDA value indicating the access method defined for the terminal. CVDA values are:

BGAM

The access method is BGAM.

BSAM

The access method is BSAM.

BTAM

The access method is BTAM.

CONSOLE

The terminal is an operating system console, accessed through MVS console support facilities.

+ NOTAPPLIC

+ The terminal is an MRO session.

TCAM

The access method is TCAM (DCB interface).

TCAMSNA

The access method is TCAM (ACB interface).

VTAM

The access method is VTAM.

| ACQSTATUS(*cvda*) (VTAM only)

| returns the same value as the TERMSTATUS option and
| is retained only for compatibility purposes. You should
| use TERMSTATUS in new applications.

ALTPAGEHT(*data-area*)

returns a halfword binary field giving the height (in lines) of the alternate page size. See also the DEFPAGEHT and PAGEHT options.

ALTPAGEWD(*data-area*)

returns a halfword binary field giving the width (in characters) of the alternate page size. See also the DEFPAGEWD and PAGEWD options.

INQUIRE TERMINAL

ALTprinter(*data-area*)

returns the 4-character name of the printer designated for print key requests and ISSUE PRINT commands from tasks at this terminal when the printer named in the PRINTER option of the TERMINAL definition is not available.

ALTPRTCOPYST(*cvda*)

returns a CVDA value indicating whether CICS is to use the hardware copy feature to satisfy a print request on the printer named in the ALTprinter option. CVDA values are:

ALTPRTCOPY

CICS is to use the hardware copy feature.

NOALTPRTCOPY

CICS is not to use the hardware copy feature.

#

APAR PQ03859

#

added the next CVDA value.

#

NOTAPPLIC

#

The terminal is not a VTAM terminal, or is a

#

remote terminal, a surrogate terminal, or a model

#

definition.

ALTSCRNHT(*data-area*)

returns a halfword binary field giving the height (in lines) of the alternate screen size. See also the DEFSCRNHT and SCRNHT options.

ALTSCRNWD(*data-area*)

returns a halfword binary field giving the width (in characters) of the alternate screen size. See also the DEFSCRNWD and SCRNWD options.

ALTSUFFIX(*data-area*)

returns the 1-character suffix that BMS appends to map set names for maps written to this terminal when the screen is the alternate size and suffixing is in use.

APLKYBDST(*cvda*)

returns a CVDA value indicating whether the terminal has the APL keyboard feature. CVDA values are:

APLKYBD

The terminal has the APL keyboard feature.

NOAPLKYBD

The terminal does not have the APL keyboard feature.

APLTEXTST(*cvda*)

returns a CVDA value indicating whether the terminal has the APL text feature. CVDA values are:

APLTEXT

The terminal has the APL text feature.

NOAPLTEXT

The terminal does not have the APL text feature.

ASCII(*cvda*)

returns a CVDA value indicating the type of ASCII code the terminal uses, if applicable. CVDA values are:

ASCII7

The code is 7-bit ASCII.

ASCII8

The code is 8-bit ASCII.

NOTAPPLIC

The terminal does not use ASCII.

ATISTATUS(*cvda*)

returns a CVDA value indicating whether CICS can initiate a task automatically (ATI) with this terminal as its principal facility.

ATI The terminal can be used in ATI.

NOATI

The terminal cannot be used in ATI.

AUDALARMST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3270 audible alarm feature. CVDA values are:

AUDALARM

The terminal has the audible alarm feature.

NOAUDALARM

The terminal does not have the audible alarm feature.

AUTOCONNECT(*cvda*)

returns a CVDA value indicating whether CICS should attempt to establish (bind) a session with this terminal when communication with VTAM is established CVDA values are:

ALLCONN

CICS binds the session. This value is returned when the AUTOCONNECT value is ALL in the associated TYPETERM definition (when you are inquiring about a terminal) or ALLCONN in the SESSIONS definition (when you are inquiring about a session).

AUTOCONN

CICS binds the session. This value is returned when the AUTOCONNECT value is YES in the associated TYPETERM definition (in an inquiry about a terminal) or AUTOCONN in the SESSIONS definition (in an inquiry about a session).

NONAUTOCONN

CICS does not bind a session.

NOTAPPLIC

+ The terminal is not a VTAM terminal, or is a
+ remote terminal, a surrogate, or a model.

BACKTRANSST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3270 background transparency feature. Background transparency allows you to control whether

the display area behind a character is clear (transparent) or shaded. CVDA values are:

BACKTRANS

The terminal has the background transparency feature.

NOBACKTRANS

The terminal does not have the background transparency feature.

COLORST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3270 extended color feature, which allows colors to be selected for individual fields or characters. CVDA values are:

COLOR

The terminal has the extended color feature.

NOCOLOR

The terminal does not have the extended color feature.

COPYST(*cvda*)

returns a CVDA value indicating whether the control unit through which the terminal is attached includes the copy feature. COPYST applies only to 3270 terminals. CVDA values are:

COPY

The control unit has the copy feature.

NOCOPY

The control unit does not have the copy feature.

CREATESESS(*cvda*) (VTAM only)

returns a CVDA value indicating whether CICS should attempt to acquire the terminal if it is required for an automatic task initiation (ATI) request. Only VTAM physical terminals can be acquired by CICS; sessions are not eligible. CVDA values are:

CREATE

The terminal can be acquired.

NOCREATE

The terminal cannot be acquired.

NOTAPPLIC

The terminal is not a VTAM terminal or is a session (APPC, LUTYPE6.1 or MRO).

| DATASTREAM(*cvda*)

| returns a CVDA value indicating the type of data stream
| used by the terminal. CVDA values are:

| DS3270

| The terminal uses the 3270 data stream.

| NOTAPPLIC

| The terminal does not use either the 3270 or SCS
| data stream.

| SCS The terminal uses SNA character strings.

DEFPAGEHT(*data-area*)

returns a halfword binary field giving the height (in lines) of the default page size. (The corresponding option in the TYPETERM definition is PAGESIZE.) See also the ALTPAGEHT and PAGEHT options.

DEFPAGEWD(*data-area*)

returns a halfword binary field giving the width (in characters) of the default page size. (The corresponding option in the TYPETERM definition is PAGESIZE.) See also the ALTPAGEWD and PAGEWD options.

DEFSCRNHT(*data-area*)

returns a halfword binary field giving the height (in lines) of the default screen size. See also the ALTSCRNHT and SCRNHT options.

DEFSCRNWD(*data-area*)

returns a halfword binary field giving the width (in characters) of the default screen size. See also the ALTSCRNWD and SCRNWD options.

DEVICE(*cvda*)

returns a CVDA value identifying the terminal or session type. CVDA values for this option are listed in "CVDA values for the DEVICE option" on page 211.

DISCREQST(*cvda*)

returns a CVDA value indicating whether CICS is to honor a request to disconnect the terminal. Disconnect requests result from an ISSUE DISCONNECT command, or a CESF (sign-off) task with the GOODNIGHT or LOGOFF option. CVDA values are:

DISCREQ

CICS will honor a request to disconnect this terminal (with a VTAM CLSDST request to terminate the session if the terminal is a VTAM terminal).

NODISCREQ

CICS will not honor a request to disconnect this terminal.

DUALCASEST(*cvda*)

returns a CVDA value indicating whether the terminal has a typewriter keyboard or an operator console keyboard. CVDA values are:

DUALCASE

The terminal has a typewriter keyboard.

NODUALCASE

+ The terminal has an operator console keyboard
+ (this keyboard is *not* restricted to a single case), or
+ is not a 3270 display.

EXITTRACING(*cvda*) (VTAM only)

returns a CVDA value indicating whether this terminal will be traced when CICS VTAM exit tracing is active (see the TCEXITSTATUS option in the INQUIRE TRACEFLAG command, on page 126). CVDA values are:

INQUIRE TERMINAL

EXITTRACE

The terminal will be traced.

NOEXITTRACE

The terminal will not be traced.

NOTAPPLIC

- + The terminal is not a VTAM terminal, or is a
- + remote terminal, a surrogate terminal, or a model
- + definition.

EXTENDEDDESST(*cvda*)

returns a CVDA value indicating whether the terminal supports the 3270 extended data stream. The terminal has this support if the TYPETERM definition specifies it either explicitly (in the EXTENDEDDES option) or implicitly, by specifying features that use the extended data stream (see the BACKTRANST, COLORST, HIGHLIGHTST, MSRCONTROLST, OUTLINEST, PARTITIONST, PROGSYMBOLST, SOSIST, and VALIDATIONST options of this command). Extended data stream support implies that the terminal accepts write structured fields commands, including QUERY, and, conversely, support for QUERY (that is, a value of ALL or COLD for the QUERY option) implies support for the extended data stream. CVDA values are:

EXTENDEDDES

The terminal supports the extended data stream.

NOEXTENDEDDES

The terminal does not support the extended data stream.

FMHPARMST(*cvda*)

returns a CVDA value indicating whether BMS accepts user-supplied values for inclusion in a function management header (FMH) to be built by BMS. This support is available only on 3650 terminals. CVDA values are:

FMHPARM

BMS allows user-supplied values.

NOFMHPARM

BMS does not allow user-supplied values.

FORMFEEDST(*cvda*)

returns a CVDA value indicating whether the terminal has the forms feed feature. CVDA values are:

FORMFEED

The terminal has the forms feed feature.

NOFORMFEED

The terminal does not have the forms feed feature.

GCHARS(*data-area*)

returns a halfword binary field giving the graphic character set global identifier (GCSGID), which identifies the set of graphic characters that can be input or output at this terminal. (The corresponding option in the TYPETERM definition is CGCSGID.)

The GCHARS option applies only to graphic terminals; for others 0 is returned.

GCODES(*data-area*)

returns a halfword binary field giving the code page global identifier (CPGID), which identifies the EBCDIC code page that defines the code points for the characters that can be input or output at the terminal. (The corresponding option in the TYPETERM definition is CGCSGID.)

The GCODES option applies only to graphic terminals; for others 0 is returned.

HFORMST(*cvda*)

returns a CVDA value indicating whether the terminal has the horizontal forms feature, which is required for use of horizontal tabbing when formatting documents for output. CVDA values are:

HFORM

The terminal has the horizontal forms feature.

NOHFORM

The device does not have the horizontal forms feature.

HIGHLIGHTST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3270 extended highlighting facility, which enables fields or characters to be displayed in reverse-video, underlined, or blinking. CVDA values are:

HIGHLIGHT

The terminal has extended highlighting.

NOHIGHLIGHT

The terminal does not have extended highlighting.

KATAKANAST(*cvda*)

returns a CVDA value indicating whether the terminal is a Katakana terminal. CVDA values are:

KATAKANA

The terminal is a Katakana terminal.

NOKATAKANA

The terminal is not a Katakana terminal.

LIGHTPENST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3270 selector pen feature. CVDA values are:

LIGHTPEN

The terminal has the selector pen feature.

NOLIGHTPEN

The terminal does not have the selector pen feature.

+ MAPNAME(*data-area*)

- + returns the 7-character name of the map that was most
- + recently referenced in the MAP option of a SEND MAP
- + command processed for this terminal. If this terminal is
- + a surrogate, and the terminal owning system is a
- + CICS/ESA 4.1 region, the map name may be the last
- + map sent by the terminal-owning region or another AOR

+ in which this terminal has been represented as a
 + surrogate device. The map name returned may no
 + longer be held in the device buffer, because an
 + intervening BMS command such as SEND TEXT or
 + SEND CONTROL (or a terminal control SEND
 + command), or operator action, may have partially or
 + completely removed the map display. If the terminal is
 + not supported by BMS (for example, this terminal is a
 + session), or CICS has no record of any map being sent,
 + the value returned is blanks.

+ **MAPSETNAME**(*data-area*)

+ returns the 8-character name of the mapset that was
 + most recently referenced in the MAPSET option of a
 + SEND MAP command processed for this terminal. If the
 + MAPSET option was not specified on the most recent
 + request, BMS uses the map name as the mapset name.
 + In both cases, the mapset name used may be suffixed
 + by a terminal or alternate suffix. If this terminal is a
 + surrogate, the mapset name may be the last mapset
 + used by the terminal owning region or another AOR in
 + which this terminal has been represented as a surrogate
 + device. If the terminal is not supported by BMS (for
 + example this terminal is a session), or CICS has no
 + record of any mapset being used, the value returned is
 + blanks.

+ **Note:** See the *CICS/ESA Application Programming
 + Guide* for information about mapset suffixing.

MODENAME(*data-area*) (**APPC only**)

returns the 8-character name of the session group to
 which the session about which you are inquiring belongs
 (from the LOGMODE option of the SESSIONS
 definition). MODENAME applies only to APPC logical
 units; for other types, the value returned is blanks.

MSRCONTROLST(*cvda*)

returns a CVDA value indicating whether the terminal
 has a magnetic slot reader. This feature is available
 only on 8775 and 3643 terminals. CVDA values are:

MSRCONTROL

The terminal has a magnetic slot reader.

NOMSRCONTROL

The terminal does not have a magnetic slot
 reader.

NATLANG(*data-area*)

| returns a 1-character value giving the national language
 | specified in the TERMINAL definition. This value cannot
 | be changed by any command, and is not necessarily the
 | same as the national language currently in use at the
 | terminal. To determine current language, see the
 | NATLANGINUSE option of the ASSIGN command in the
 | *CICS/ESA Application Programming Reference* manual.
 | Possible values are:

| E English
 | K Katakana

| blank

| No value specified

NATURE(*cvda*)

returns a CVDA value identifying the nature of the
 terminal definition. CVDA values are:

MODEL

A model terminal definition, used only to build
 definitions of real terminals.

SESSION

A remote session.

SURROGATE

A surrogate terminal definition, representing a
 terminal owned by another CICS region.

TERMINAL

A physical terminal definition.

NETNAME(*data-area*)

| returns the 8-character network name of the terminal
 | about which you are inquiring.

| For a physical terminal, this is the name by which this
 | terminal is known to VTAM. For ISC sessions, it is the
 | name by which the session (or session group, if there
 | are parallel sessions) is known to VTAM. For MRO
 | sessions, it is the name used by the connected region to
 | log on to the interregion communication program.

| **Note:** The description above applies to the NETNAME
 | option in an INQUIRE TERMINAL command. In an
 | INQUIRE NETNAME command, the roles of NETNAME
 | and TERMINAL are reversed. NETNAME specifies the
 | name of the terminal or session about which you are
 | inquiring to CICS, rather than returning information, and
 | TERMINAL returns the corresponding terminal identifier
 | if you use it. See the description of INQUIRE
 | NETNAME on page 85.

NEXTTRANSID(*data-area*)

returns the 4-character identifier of the transaction to be
 executed to process the next unsolicited input from this
 terminal. This value comes from the TRANSACTION
 value in the TERMINAL or SESSIONS definition, if one
 has been specified. If the value has not been specified,
 it was set by the previous task for which the terminal
 was principal facility (in the TRANSID option of its final
 RETURN command) and will be blanks if that task did
 not specify a value or if an active task has the terminal
 as principal facility.

OBFORMATST(*cvda*)

returns a CVDA value indicating whether outboard
 formatting can be used for this terminal. CVDA values
 are:

NOOBFORMAT

+ This terminal does not support outboard
 + formatting.

OBFORMAT

+ This terminal supports outboard formatting.

INQUIRE TERMINAL

OBOPERIDST(*cvda*)

returns a CVDA value indicating whether CICS uses outboard operator identifiers to support the BMS routing facilities at this terminal. This option only applies to the 3790 and 3770 batch data interchange logical units. CVDA values are:

NOOOPERID

CICS does not use outboard operator identifiers.

OBOPERID

CICS uses outboard operator identifiers.

OPERID(*data-area*)

returns the 3-character operator identification code of the user signed on at the terminal.

Note: If the terminal is a surrogate terminal, this value may not be current; it represents the user signed on at the time the terminal definition was shipped from the owning CICS region to this one, who may since have signed off. For information, see the *CICS/ESA CICS-RACF Security Guide*.

+

APAR PQ01572

+

added the following note

+

The OPERID may also be different from that of the user currently signed on if it has been changed with the SET TERMINAL command.

+

+

+

OUTLINEST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3270 field outlining feature. CVDA values are:

NOOUTLINE

The terminal does not support field outlining. (This value is always returned for a model terminal.)

OUTLINE

The terminal supports field outlining.

PAGEHT(*data-area*)

returns a halfword binary field giving the height (in lines) of the current page size for the terminal. See the DEFPAGEHT and ALTPAGEHT options.

PAGESTATUS(*cvda*)

returns a CVDA value indicating how pages of BMS messages with a disposition of PAGING should be delivered to the terminal. CVDA values are:

AUTOPAGEABLE

Pages are written automatically in sequence.

PAGEABLE

Pages are written on request from the operator.

PAGEWD(*cvda*)

returns a halfword binary field giving the width (in characters) of the current page size for the terminal. See also the DEFPAGEWD and ALTPAGEWD options.

PARTITIONSST(*cvda*)

returns a CVDA value indicating whether the terminal supports partitions. CVDA values are:

NOPARTITIONS

The terminal does not support partitions.

PARTITIONS

The terminal supports partitions.

PRINTADAPTST(*cvda*)

returns a CVDA value indicating whether the terminal has the printer adapter feature. CVDA values are:

NOPRINTADAPT

The terminal does not have a printer adapter.

PRINTADAPT

The terminal has a printer adapter.

PRINTER(*data-area*)

returns the 4-character name of the preferred printer for print key requests and ISSUE PRINT commands from tasks at this terminal. This printer is used if available; if not, the printer named in the ALTPRINTER option is second choice.

PROGSYMBOLST(*cvda*)

returns a CVDA value indicating whether the terminal supports the 3270 programmed symbol feature, which enables the terminal to use multiple character sets. CVDA values are:

NOPROGSYMBOL

The terminal does not support programmable symbols.

PROGSYMBOL

The terminal supports programmable symbols.

PRTCOPYST(*cvda*)

returns a CVDA value indicating whether CICS is to use the hardware copy feature to satisfy a print request on the printer named on the PRINTER option. CVDA values are:

NOPTCOPY

CICS is not to use the hardware copy feature.

#

APAR PQ03859

#

added the next CVDA value.

#

NOTAPPLIC

#

The terminal is not a VTAM terminal, or is a

#

remote terminal, a surrogate terminal, or a model definition.

#

PRTCOPY

CICS is to use the hardware copy feature.

QUERYST(*cvda*)

returns a CVDA value indicating whether and when CICS should use a QUERY structured field to determine the characteristics of the terminal.

ALLQUERY

The terminal is to be queried each time it is connected.

COLDQUERY

The terminal is to be queried only when it is first connected after a cold start of CICS. The device characteristics are stored on the global catalog for use on subsequent warm and emergency starts.

NOQUERY

The terminal is not to be queried.

RELREQST(*cvda*) (VTAM only)

returns a CVDA value indicating whether CICS is to honor requests from VTAM to release the terminal or session. CVDA values are:

NORELREQ

CICS cannot release the logical unit, or the access method is not VTAM.

RELREQ

CICS can release the logical unit.

REMOTENAME(*data-area*)

returns the 4-character name of this terminal in the remote CICS region in which it is defined. REMOTENAME applies only to terminals defined as remote; for others the value returned is blanks.

REMOTESYSTEM(*data-area*)

returns the 4-character name of the CICS region in which the terminal is defined. REMOTESYSTEM applies only to terminals defined as remote; for others the value returned is blanks.

SCRNHT(*data-area*) (or SCREENHEIGHT)

returns a halfword binary field giving the height (in lines) of the current screen size. See also the DEFSCRNHT and ALTSCRNHT options.

Note: SCRNHT is a synonym for the SCREENHEIGHT option of earlier releases of CICS. For compatibility, CICS recognizes SCREENHEIGHT as equivalent.

SCRNWD(*data-area*) (or SCREENWIDTH)

returns a halfword binary field giving the current width of the terminal screen (in characters). See the DEFSCRNWD and ALTSCRNWD options.

Note: SCRNWD is a synonym for the SCREENWIDTH option of earlier releases of CICS. For compatibility, CICS recognizes SCREENWIDTH as equivalent.

SECURITY(*cvda*)

returns a CVDA value indicating whether the terminal has preset security, that is, whether a USERID value has been specified in the TERMINAL or SESSIONS definition, so that it is permanently signed on. CVDA values are:

NOPRESETSEC

The terminal does not have preset security.

PRESETSEC

The terminal has preset security.

SERVSTATUS(*cvda*)

returns a CVDA value indicating whether the terminal is available for use (from the point of view of the local CICS system, which may be different from the system which owns the terminal). SERVSTATUS corresponds to the INSERVICE option in the TERMINAL definition. "Available" (INSERVICE) does not necessarily imply, for a VTAM terminal, that the terminal is acquired.

GOINGOUT

The terminal will be put in OUTSERVICE status as soon as some current work has completed and is not available to new tasks.

INSERVICE

The terminal is available.

OUTSERVICE

The terminal is not available.

SESSIONTYPE(*cvda*)

returns a CVDA value identifying the type of the session about which you are inquiring. This option applies only to VTAM sessions. CVDA values are:

APPCPARALLEL

This is a parallel APPC session group.

APPCSINGLE

This is a single APPC session.

LU61

This is an LUTYPE6.1 session.

NOTAPPLIC

The terminal is not one of the above.

SIGNONSTATUS(*cvda*)

returns a CVDA value identifying whether the terminal currently has a signed-on user. CVDA values are:

SIGNEDOFF

The terminal does not have a signed-on user.

SIGNEDON

The terminal has a signed-on user.

SOSIST(*cvda*)

returns a CVDA value indicating whether the terminal supports mixed EBCDIC and double-byte character set (DBCS) fields. CVDA values are:

NOSOSI

The terminal does not support mixed fields.

SOSI

The terminal supports mixed fields.

TASKID(*data-area*)

returns a fullword binary field giving the number of the user task currently executing at this terminal. Zero is returned if no task is using the terminal.

INQUIRE TERMINAL

TCAMCONTROL(*data-area*) (TCAM only)

returns a 1-character TCAM control byte giving which segment of a message has passed between CICS and TCAM.

The meanings are:

- 00 Null
- 40 Intermediate part of message
- F1 First part of message
- F2 Last part of message
- F3 Whole message
- F4 Intermediate part of message, end of record
- F5 First part of message, end of record
- F6 Last part of message, end of record
- F7 Whole message, end of record
- FE TCAM is not active
- FF Not applicable (non-TCAM terminal)

TERMINAL(*data-value*)

specifies the 4-character name of terminal or session about which you are inquiring, in an INQUIRE TERMINAL command. In an INQUIRE NETNAME command, this option *returns* the terminal identifier that corresponds to the NETNAME value you specified. See the NETNAME option and the general information for this command.

TERMMODEL(*data-area*)

returns a halfword binary field giving the terminal model number.

TERMPRIORITY(*data-area*)

returns a fullword binary field giving the priority of the terminal relative to other terminals, in the range 0–255.

| TERMSTATUS(*cvda*) (VTAM only)

| returns a CVDA value indicating whether CICS is in session with the logical unit represented by this terminal. CVDA values are:

- | ACQUIRED
| CICS is in session with the logical unit.
- | ACQUIRING
| The session is in the process of being acquired.
- | NOTAPPLIC
| The terminal is not a VTAM terminal.
- | RELEASED
| CICS is not in session with the logical unit.
- | RELEASING
| The session is in the process of being released.

TEXTKYBDST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3270 text-keyboard feature. CVDA values are:

NOTEXTKYBD

The terminal does not have the text-keyboard feature.

TEXTKYBD

The terminal has the text-keyboard feature.

TEXTPRINTST(*cvda*)

returns a CVDA value indicating whether the terminal has the 3288 text-print feature. CVDA values are:

NOTEXTPRINT

The terminal does not have the text-print feature.

TEXTPRINT

The terminal has the text-print feature.

TRACING(*cvda*)

+ returns a CVDA value indicating the type of tracing defined for this terminal. CVDA values are:

+ SPECTRACE

+ Special tracing is specified.

+ STANTRACE

+ Standard tracing is specified.

+ For a task that has this terminal as its principal facility, this value is combined with the TRACING option value of the transaction the task is executing to determine whether tracing is standard, special, or suppressed.

+ If the transaction TRACING value is SUPPRESSED, no tracing occurs. Otherwise, tracing is special if either the terminal or the transaction specifies SPECTRACE, standard if both specify STANTRACE.

+ A TRACING value of STANTRACE is assigned when the terminal is defined. You can specify SPECTRACE only with a SET TERMINAL command or the CICS-supplied CETR transaction.

TRANSACTION(*data-area*)

+ returns the 4-character identifier of the transaction being executed by the task for which this terminal is the principal facility. Blanks are returned if no task is currently running at the terminal.

TTISTATUS(*cvda*)

returns a CVDA value indicating whether this terminal can initiate tasks by entering unsolicited input. CVDA values are:

NOTTI

This terminal cannot initiate transactions.

TTI This terminal can initiate transactions.

UCTRANST(*cvda*)

returns a CVDA value indicating whether input from this terminal is translated to upper case automatically, at the time of receipt. (Translation can be suppressed, but only in a conversational task, when input is solicited with a RECEIVE or CONVERSE ASIS command.) This value comes from the UCTRAN option of the TYPETERM definition associated with the terminal; there

is also an UCTRAN option in a PROFILE definition, but that value is not pertinent here. CVDA values are:

NOUCTRAN

Input from this terminal is not translated to upper case on receipt. (It will be translated before presentation to the task issuing a RECEIVE, however, if the PROFILE definition for the transaction being executed specifies translation.)

TRANIDONLY

This value is the same as NOUCTRAN, with one difference. If the input is unsolicited, and CICS needs to use the initial characters of the input to decide which transaction to execute, that decision is made from a *copy* of the input which has been translated to upper case. There is no difference in the data presented to the task between these two options.

UCTRAN

The input is translated to upper case on receipt (and unaffected by the translation option in the PROFILE).

USERAREA(*ptr-ref*)

| returns the address of the terminal control table user
| area (TCTUA) for this terminal. If there is no TCTUA,
| the address returned is X'FF000000'.

USERAREALEN(*data-area*)

| returns a halfword binary field giving the length of the
| user area. Zero is returned if there is no user area.

USERID(*data-area*)

returns the 8-character identifier of the user signed on at this terminal or session.

+ If there is no signed-on user, the default userid—as
+ specified in the DFLTUSER system initialization
+ parameter—is returned.

USERNAME(*data-area*)

returns the 20-character name of the user signed on at this terminal or session (that is, the name corresponding to the USERID option value). If the information, which is provided by the external security manager, is shorter than 20 bytes, CICS pads it to 20 with trailing blanks. Blanks are returned if there is no signed on user.

VALIDATIONST(*cvda*)

returns a CVDA value identifying whether the device has the extended validation feature, which allows you to request special processing of keyboard input, additional to normal 3270 function. This feature is available only on 8775 and 3290 terminals. CVDA values are:

NOVALIDATION

The terminal does not have the extended validation feature or is a model terminal.

VALIDATION

The terminal has the extended validation feature.

VFORMST(*cvda*)

returns a CVDA value indicating whether the terminal has the vertical forms feature, which is required for use of vertical tabbing when formatting documents for output. CVDA values are:

NOVFORM

The device does not have the vertical forms feature.

VFORM

The terminal has the vertical forms feature.

ZCPTRACING(*cvda*) (VTAM only)

returns a CVDA value indicating whether this terminal will be traced when CICS tracing for VTAM terminals is turned on. CVDA values are:

NOTAPPLIC

The terminal is not a VTAM terminal, or is a surrogate terminal or a model definition.

NOZCPTRACE

The terminal will not be traced.

ZCPTRACE

The terminal will be traced.

Conditions**END**

RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

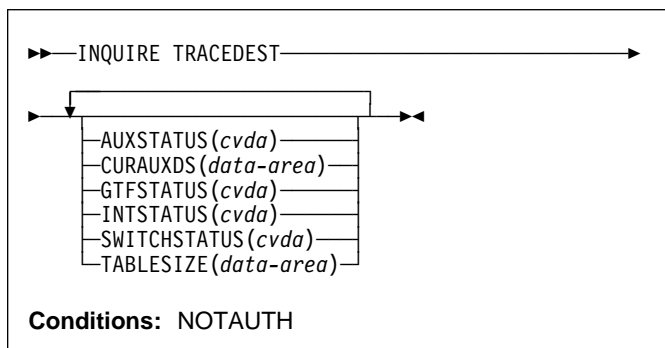
TERMDERR

RESP2 values:

1 The named terminal cannot be found.

INQUIRE TRACEDEST

Retrieve information about tracing.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

The INQUIRE TRACEDEST command tells you where CICS trace entries are currently being written. There are three possible destinations, which can be used in any combination: the CICS internal trace table, the auxiliary trace data set, and the MVS Generalized Trace Facility (GTF). The number and types of trace entries are controlled by switch settings that you can determine with the INQUIRE TRACEFLAG and INQUIRE TRACETYPE commands.

Options

AUXSTATUS(*cvda*)

returns a CVDA value indicating whether auxiliary tracing is active, that is, whether trace entries are being written to an auxiliary trace data set. CVDA values are:

AUXPAUSE

Auxiliary tracing is not currently active, but was earlier in the current execution of CICS. It was suspended with a SET TRACEDEST AUXPAUSE command (or the CEMT equivalent). The current auxiliary trace data set has been left open, and a subsequent SET TRACEDEST AUXSTART command will cause trace entries to be written immediately following those that were written before the AUXPAUSE request.

AUXSTART

Auxiliary tracing is active.

AUXSTOP

Auxiliary tracing is not active (the current trace data set, if any, is closed).

CURAUXDS(*data-area*)

returns the 1-character identifier of the current auxiliary trace data set, which can be 'A', 'B' or blank.

If your CICS system is initialized to allow auxiliary tracing, it will have either a single auxiliary trace data set, known as the 'A' data set, or two, 'A' and 'B'. The "current" or "active" one receives trace entries when auxiliary tracing is turned on, and the other, if there are two, is a standby, for use when the current one becomes full (see the SWITCHSTATUS option). If there is no auxiliary trace data set, the CURAUXDS value is blank.

GTFSTATUS(*cvda*)

returns a CVDA value indicating whether GTF tracing is active, that is, whether CICS is directing trace entries to the MVS Generalized Trace Facility (GTF). CVDA values are:

GTFSTART

GTF tracing is active.

GTFSTOP

GTF tracing is not active.

Note: In order to record trace entries on GTF, CICS must be initialized with GTF support (in the GTFTR system initialization option), GTF tracing must be started (with a SET TRACEDEST GTFSTART command or equivalent), and GTF trace must be started in MVS with the TRACE=USR option. If either of the first two conditions is not met, GTFSTATUS will be GTFSTOP. However, GTFSTATUS can be GTFSTART without the third condition; in this case, no entries are written to GTF, but there is no other error indication.

INTSTATUS(*cvda*)

returns a CVDA value indicating whether internal tracing is active, that is, whether trace entries are being written in the internal trace table. CVDA values are:

INTSTART

Internal tracing is on.

INTSTOP

Internal tracing is off.

Note: Exception trace entries are always written to the internal trace table, regardless of the INTSTATUS value.

SWITCHSTATUS(*cvda*)

returns a CVDA value indicating the action that CICS is to take when the active auxiliary trace data set fills. If there are two data sets, CICS can switch them automatically when this occurs. Switching involves closing the current active data set, opening the standby, and reversing the designation of which is active and standby. Without automatic switching, auxiliary tracing is stopped and cannot resume without a SET TRACEDEST command or the CEMT equivalent.

CVDA values are:

NOSWITCH

CICS takes no action.

SWITCHALL

CICS is to switch data sets every time the current one is full.

SWITCHNEXT

CICS is to switch data sets when the current one is full, but only once; thereafter NOSWITCH will be in effect.

TABLESIZE(*data-area*)

returns a fullword binary field giving the size of the internal trace table in kilobytes.

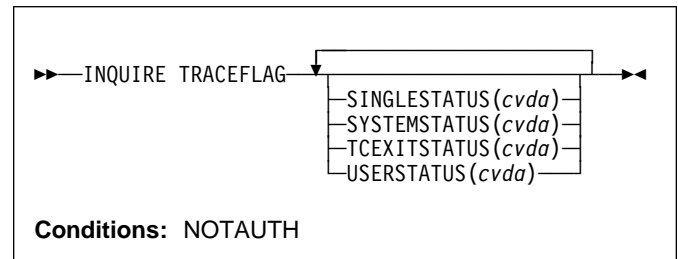
Conditions**NOTAUTH**

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

INQUIRE TRACEFLAG

Retrieve information about trace flags.



For more information about the use of CVDA, see “CICS-value data areas (CVDA)” on page 7.

Description

The INQUIRE TRACEFLAG command returns the current settings of the flags that control tracing in CICS generally and for the task that issued the command specifically.

Tracing facilities and control are discussed in detail in the *CICS/ESA Problem Determination Guide*.

Options**SINGLESTATUS**(*cvda*)

returns a CVDA value indicating whether tracing is turned on or suppressed for the task that issued this INQUIRE TRACEFLAG command. No non-exception trace entries are made for a task when this flag is off, regardless of the settings of the master trace flags (exception trace entries are *always* recorded).

- + The SINGLESTATUS value comes from the TRACE
- + option in the definition of the TRANSACTION the task is
- + executing, unless a different value has been specified,
- + either for the transaction or for the terminal that is the
- + principal facility, by means of the CICS-supplied CETR
- + transaction. When a task is in progress, its
- + SINGLESTATUS value can also be changed with a SET
- + TRACEFLAG command.

CVDA values are:

SINGLEOFF

Tracing is suppressed.

SINGLEON

Tracing is allowed.

SYSTEMSTATUS(*cvda*)

returns a CVDA value indicating the status of the system master trace flag. This flag governs whether CICS makes or suppresses standard trace entries (it does not affect special or exception trace entries). It applies to all tasks and all system activity; however, for such trace entries to be recorded for any particular task, both the

INQUIRE TRACETYPE

system master flag and the SINGLESTATUS flag for that task must be on.

CVDA values are:

SYSTEMOFF

Standard tracing is suppressed.

SYSTEMON

Standard tracing is active.

TCEXITSTATUS(*cvda*) (VTAM only)

+ returns a CVDA value indicating which invocations of the CICS VTAM exits are being traced.

+ Two types of exit activity can be traced: invocations associated with particular terminals that have been designated for VTAM exit tracing (“terminal-specific” activity) and invocations not related to any particular terminal (“nonterminal-specific” activity).

CVDA values are:

NOTAPPLIC

VTAM is not installed in the system.

TCEXITALL

All exit activity is being traced.

TCEXITNONE

No exit activity is being traced.

TCEXITSYSTEM

+ Nonterminal-specific activity is being traced, but terminal-specific activity is not.

USERSTATUS(*cvda*)

returns a CVDA value indicating the status of the user master trace flag. This flag governs whether non-exception user trace entries are recorded or suppressed (entries that specify the EXCEPTION option are never suppressed). It applies to all tasks; however, for such entries to be recorded for any particular task, both the user master trace flag and the SINGLESTATUS flag for that task must be on. CVDA values are:

USEROFF

User tracing is suppressed.

USERON

User tracing is allowed.

Conditions

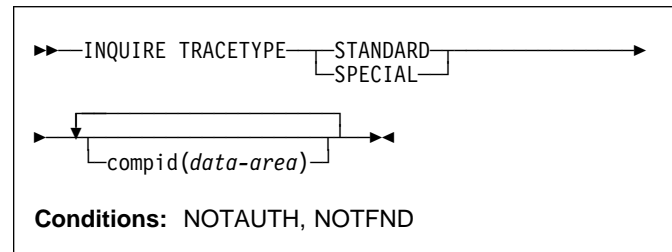
NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

INQUIRE TRACETYPE

Retrieve information about CICS system tracing.



Description

The INQUIRE TRACETYPE command tells you which levels of tracing are currently in effect for particular CICS system components.

Each CICS component has trace levels defined separately for standard CICS tracing and special tracing (see the *CICS/ESA Problem Determination Guide* for definitions of these terms and for information about CICS tracing in general). You can ask about either type in an INQUIRE TRACETYPE command, and for any number of components, but you can ask about only one type in a single command.

For each component that you specify, the trace levels are returned as a 32-bit string (in a 4-character data area). The bits are read from left to right; that is, the first bit is on if trace level 1 is in effect, bit 2 corresponds to trace level 2, and so on. A value of X'80000000' thus represents trace level 1. Bits for trace levels that are not defined are set to zero.

Options

compid(*data-area*)

returns the trace levels for the CICS component identified by “compid” in the format described above.

CICS components can be identified by a 2-character identifier or, in some cases, a descriptive keyword. For example, to determine the trace levels for the directory manager component of CICS, you can specify either:

```
INQUIRE TRACETYPE DD(data-area)
```

or

```
INQUIRE TRACETYPE DIRMGR(data-area)
```

The following list shows all the 2-character identifiers, and the keywords for those components that have them.

AP	APPLICATION	Application
BF		Built-in functions
BM		Basic mapping support
CP	CPI	Common programming interface

	DC	Dump control
	DD	DIRMGR Directory manager
	DI	Batch data interchange
	DM	DOMAINMGR Domain manager
	DS	DISPATCHER Dispatch manager
	DU	DUMP Dump manager
	EI	EXEC interface
	FC	File control and DL/I
	GC	GLOBALCATLG CICS global catalog manager
	IC	Interval control
	IS	Inter-system communication
	JC	Journal control
	KC	Task control
	KE	KERNEL Kernel
	LC	LOCALCATLG CICS local catalog manager
	LD	LOADER Program load manager
	LM	LOCKMGR Lock manager
	ME	MESSAGE Message manager
	MN	MONITOR Monitoring manager
	PA	PARAMGR Parameter manager
	PC	Program control
	PG	PROGMGR Program manager
	SC	Storage control
	SM	STORAGE Storage manager
	SP	Syncpoint manager
	ST	STATISTICS Statistics manager
	SZ	Front-end programming interface
	TC	Terminal control
	TD	Transient data
	TI	TIMER Timer manager
	TR	TRACE Trace manager
	TS	Temporary storage
	UE	User exit interface
	US	USER User interface
	XM	TRANMGR Transaction manager
	XS	SECURITY Security manager

SPECIAL

indicates that CICS should return the trace levels for special tracing.

STANDARD

indicates that CICS should return the trace levels for standard tracing.

Conditions

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

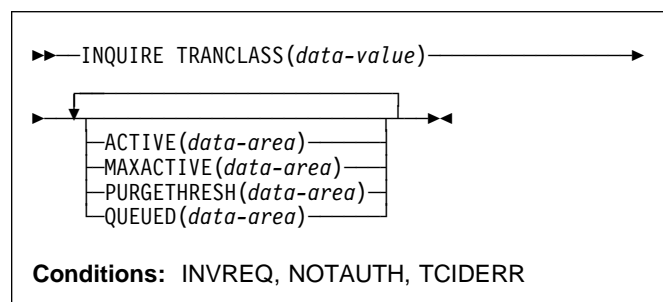
NOTFND

RESP2 values:

- + **1** CICS was initialized without support for at least one of the components listed in the command;
- + trace levels were returned for all other components.

INQUIRE TRANCLASS

Retrieve information about a transaction class.



Description

The INQUIRE TRANCLASS command allows you to determine the limits defined for a transaction class and the current activity within the class.

Browsing

You can also browse through the definitions of all the transaction classes in your system by using the browse options (START, AT, NEXT and END) on INQUIRE TRANCLASS commands. In browse mode, definitions are returned in alphabetical order, and you can specify a starting point with the AT option if you wish. See "Browsing resource definitions" on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

ACTIVE(data-area)

returns a fullword binary field giving the current number of tasks in this class. This count does not include tasks that are queued waiting for initial dispatch.

MAXACTIVE(data-area)

returns a fullword binary field giving the largest number of tasks in the transaction class which are allowed to run concurrently.

PURGETHRESH(data-area)

returns a fullword binary field giving the maximum number of tasks in this class that can be queued awaiting initial dispatch (see the QUEUED option). Tasks in this class that arrive while the queue is at its PURGETHRESH limit are purged.

QUEUED(data-area)

returns a fullword binary field giving the number of tasks that are queued awaiting initial dispatch. Queuing occurs either because the number of active tasks is already at the maximum, or because the maximum for the system has been reached (see the MAXTASKS option in the INQUIRE SYSTEM command).

INQUIRE TRANDUMPCODE

TRANCLASS(*data-value*)

specifies the 8-character name of the transaction class about which you are inquiring. If the class is one of the numbered classes used in earlier releases of CICS, its name is DFHTCLnn, where "nn" is the two-digit class number.

Conditions

INVREQ

RESP2 values:

12 The TRANCLASS definition is in use.

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

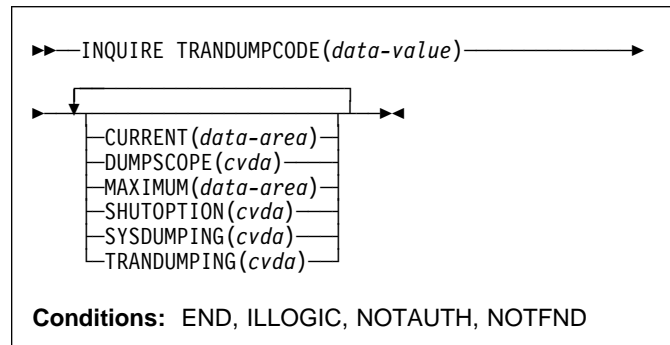
TCIDERR

RESP2 values:

1 The transaction class cannot be found.

INQUIRE TRANDUMPCODE

Retrieve information about a transaction dump code.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

The INQUIRE TRANDUMPCODE command allows you to look at some of the information in the transaction dump table entry for a particular transaction dump code.

The table entry tells CICS what actions to take when a transaction dump request with this code is received.

Possible actions are: taking a transaction dump, taking a system dump (an MVS SDUMP), forwarding an SDUMP + request to related MVS images, and shutting down CICS.

The table entry also indicates how many times this set of actions is to be taken (the MAXIMUM option); requests received after the maximum are counted (the CURRENT option), but otherwise ignored.

CICS provides a transaction dump table with default actions for CICS transaction abend codes (those beginning with the letter A). These can be changed and others can be added with the SET TRANDUMPCODE command or the CEMT transaction; such changes are preserved over executions of CICS, until a cold start occurs.

+ CICS builds table entries, using default values, when it + receives a dump request with a code for which it does not + have an entry. You can also add your own entries with the + SET TRANDUMPCODE command or a CEMT transaction.

+ Entries you add remain over executions of CICS until a cold + start occurs, but the entries CICS builds are considered + temporary and are discarded at shutdown.

+ Consequently, if you enquire about a code that is not + explicitly defined before it appears in a dump request, you + will get a "not found" response.

Browsing

You can also browse through all of the entries in the transaction dump table by using the browse options (START, NEXT and END) on INQUIRE TRANDUMPCODE commands. See "Browsing resource definitions" on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

CURRENT(*data-area*)

returns a fullword binary field giving the number of dump requests with this dump code made since the count was last reset. (The count is reset automatically at CICS shutdown and can be reset explicitly with a SET SYSDUMPCODE RESET command or its CEMT equivalent.) The count includes requests that do not result in dumps, either because they are suppressed for this code or because the number for this code has reached its maximum.

DUMPSCOPE(*cvda*)

returns a CVDA value indicating whether a request for a dump with this dump code should cause an SDUMP (system dump) request to be sent to related MVS images.

A related image is one which contains a CICS region doing work on behalf of the task that caused the dump request—specifically, a region that has a task doing work under the same APPC token as the task causing the dump.

The sending of SDUMP requests occurs only when the table entry for this code specifies a system dump (that is, the SYSDUMPING value is SYSDUMP), and only in a sysplex environment executing under MVS/ESA 5.1 and the MVS workload manager.

CVDA values are:

LOCAL

SDUMP requests are not to be sent.

RELATED

SDUMP requests are to be sent.

MAXIMUM(*data-area*)

returns a fullword binary field giving the maximum number of times CICS will take the set of actions indicated in the transaction dump table entry when a dump request with this code is received. A value of 999 means

APAR PQ01620
 # changed this sentence

the default, 'no limit'.

SHUTOPTION(*cvda*)

returns a CVDA value indicating whether the CICS system is to be shut down after a request for a dump with this dump code. CVDA values are:

NOSHUTDOWN

The CICS system is not to shut down.

SHUTDOWN

The CICS system is to shut down.

SYSDUMPING(*cvda*)

returns a CVDA value indicating whether a system dump should be taken when a transaction dump request with this code is received. Even when the dump table entry specifies a system dump, however, one will be taken only when the CURRENT value is no greater than the MAXIMUM, and system dumps are not suppressed system-wide (see the DUMPING option in the INQUIRE SYSTEM command). CVDA values are:

NOSYSDDUMP

A system dump is not to be taken.

SYSDUMP

A system dump is to be taken.

TRANDUMPCODE(*data-value*)

specifies the 4-character transaction dump code about which you are inquiring. A valid transaction dump code has no leading or imbedded blanks.

TRANDUMPING(*cvda*)

returns a CVDA value indicating whether a transaction dump should be taken when a transaction dump request with this code is received. Even when the dump table entry specifies a transaction dump, however, one will be taken only when the CURRENT value is no greater than the MAXIMUM. CVDA values are:

NOTRANDUMP

The transaction dump is to be suppressed.

TRANDUMP

The transaction dump is to be taken.

Conditions

END

RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

INQUIRE TRANSACTION

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

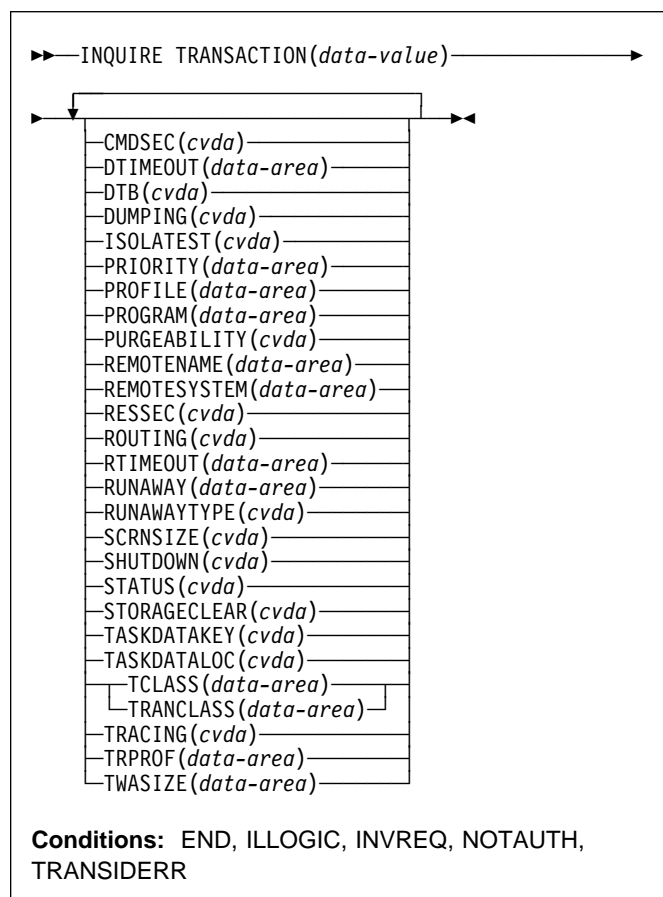
NOTFND

RESP2 values:

- 1 The dump code cannot be found.

INQUIRE TRANSACTION

Retrieve information about a TRANSACTION definition.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

For more information about device properties, see the # TYPETERM chapter in the *CICS/ESA Resource Definition Guide*.

Description

The INQUIRE TRANSACTION command retrieves information about a particular transaction installed in your CICS system.

Most of the values come from the TRANSACTION resource definition, but a few come from the PROFILE definition to which it points (these are noted in the descriptions). See the *CICS/ESA Resource Definition Guide* for full details about the attributes of these two types of resources.

Many of the values produced by an INQUIRE TRANSACTION command are the same as those produced by the same-named options in an INQUIRE TASK command, when the task is executing the transaction, because a task acquires most of its characteristics from the definition of the

transaction. However, as noted in the description of that command, the values for a task also reflect the CICS system environment.

Furthermore, when a task is routed from one CICS to another, the transaction specified in the sending region may be different from the one executed in the receiving region, so that an inquiry about its TRANSACTION value can produce different results in the sending and receiving regions. Indeed, in the case of dynamic routing, the transaction specified in the sending CICS (and shown as the TRANSACTION value in an INQUIRE TASK there) need not even be defined if the default processing for an undefined transaction code is dynamic routing.

Browsing

You can also browse through all of the TRANSACTION definitions in your system by using the browse options (START, AT, NEXT, and END) on INQUIRE TRANSACTION commands. In browse mode, the definitions are returned in alphabetic order, and you can specify a starting point with the AT option if you wish. See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

CMDSEC(*cvda*)

returns a CVDA value indicating whether command security checking should be performed for tasks executing this transaction. CVDA values are:

```
# CMDSECNO
#     Command security checking should not be
#     performed.
# CMDSECYES
#     Command security checking should be performed.
```

DTB(*cvda*)

returns a CVDA value indicating how uncommitted changes made to recoverable resources by a task executing this transaction should be handled if the task fails. CVDA values are:

```
BACKOUT
    Uncommitted changes should be backed out.
COMMIT
    Uncommitted changes should be committed.
WAIT
    The disposition of uncommitted changes should be
    deferred until further information is available.
```

DTIMEOUT(*data-area*)

returns a fullword binary field giving the deadlock time-out value (in seconds) for a task executing this transaction. CICS abends a task that waits for a locked resource longer than its deadlock timeout value.

DUMPING(*cvda*)

returns a CVDA value indicating whether CICS should take a transaction dump if a task executing this transaction terminates abnormally. CVDA values are:

```
NOTRANDUMP
    No dump should be taken.
TRANDUMP
    A dump should be taken.
```

This value applies only to abend dumps and has no effect on DUMP TRANSACTION commands.

ISOLATEST(*cvda*)

returns a CVDA value indicating whether a task executing this transaction should run isolated when isolation is active in the system.

Isolation limits the access, for both read and write, of user-key programs to task storage. A program executing in user key on behalf of an isolated task can access the task storage of only that task, and this storage cannot be accessed by programs executing in user key on behalf of other tasks. Isolation does not affect access by CICS-key programs and does not apply to storage with the SHARED attribute or any other non-task storage.

Isolation must be turned on for the system as well as the transaction in order for a task to run isolated. (See the TRANISOLATE option of the INQUIRE SYSTEM command.) CVDA values are:

```
ISOLATE
    Tasks should run isolated.
NOISOLATE
    Tasks should not run isolated.
```

PRIORITY(*data-area*)

returns a fullword binary field giving the priority of this transaction relative to other transactions in the CICS system, in the range 1–255.

PROFILE(*data-area*)

returns the 8-character name of the PROFILE definition for this transaction. The profile defines attributes that govern the interaction between a task executing the transaction and the terminal or session which is its principal facility.

PROGRAM(*data-area*)

returns the 8-character name of the first program invoked by a task executing this transaction.

PURGEABILITY(*cvda*)

```
+ returns a CVDA value indicating whether CICS is
+ allowed to purge this task (that is, to terminate it
+ abnormally). Purge requests come from SET TASK
+ PURGE commands (or CEMT equivalents), and CICS
+ can generate them internally to reclaim resources to
+ relieve a system stall condition. CVDA values are:
```

INQUIRE TRANSACTION

NOTPURGEABLE

The task cannot be purged.

PURGEABLE

The task can be purged.

The PURGEABILITY value is set initially by the SPURGE option in the definition of the TRANSACTION this task is executing.

REMOTENAME(*data-area*)

returns the 8-character name by which this transaction is known in the remote system, if it is defined as a remote transaction. (Read the description of "Defining a TRANSACTION" in the *CICS/ESA Resource Definition Guide* for a fuller discussion of the length of REMOTENAME). Blanks are returned if the transaction is not remote.

REMOTESYSTEM(*data-area*)

returns the 4-character name of the remote system on which this transaction is defined, if it is defined as a remote transaction. Blanks are returned if the transaction is not remote.

RESSEC(*cvda*)

returns a CVDA value identifying whether resource-level security checking should be performed for a task executing this transaction. CVDA values are:

RESSECNO

Resource-level checking should not be performed.

RESSECYES

Resource-level checking should be performed.

ROUTING(*cvda*)

returns a CVDA value indicating whether a task executing this transaction is subject to dynamic routing. CVDA values are:

DYNAMIC

The task can be routed dynamically.

STATIC

The task cannot be routed dynamically.

RTIMEOUT(*data-area*)

returns a fullword binary field giving the read time-out value for a task executing this transaction, in seconds. CICS abends a task if it waits for input longer than its read time-out value. This value is defined in the PROFILE definition (see the PROFILE option).

RUNAWAY(*data-area*)

returns a fullword binary field giving the "runaway task" time, in milliseconds, for tasks executing this transaction. If a task keeps control of the processor for more than this interval, CICS assumes it is in a loop and abends it. If the value is zero, CICS does not monitor the task for a runaway condition.

RUNAWAYTYPE(*cvda*)

returns a CVDA value indicating the source of the RUNAWAY option value for this transaction. CVDA values are:

SYSTEM

The value is the current default for the system (see the ICVR option of the INQUIRE SYSTEM command).

USER

The value was defined explicitly in the TRANSACTION definition.

SCRNSIZE(*cvda*)

returns a CVDA value indicating whether a task executing this transaction should use the alternate or the default screen size. This value is defined in the PROFILE definition (see the PROFILE option). CVDA values are:

ALTERNATE

The alternate screen size is to be used.

DEFAULT

The default screen size is to be used.

SHUTDOWN(*cvda*)

returns a CVDA value indicating whether this transaction can be executed during CICS shutdown by a task created to process unsolicited input. (the transaction also can be executed in this situation if it appears in the transaction list table (XLT) for shutdown). CVDA values are:

SHUTDISABLED

The transaction cannot be executed.

SHUTENABLED

The transaction can be executed.

STATUS(*cvda*)

returns a CVDA value indicating whether the transaction is available for use. CVDA values are:

DISABLED

The transaction is not available for use.

ENABLED

The transaction is available for use.

STORAGECLEAR(*cvda*)

returns a CVDA value indicating whether CICS should clear storage that is released from a task executing this transaction (to prevent other tasks accidentally viewing confidential data). CVDA values are:

CLEAR

Storage will be cleared.

NOCLEAR

Storage will not be cleared.

TASKDATAKEY(*cvda*)

returns a CVDA value indicating the key of the storage CICS assigns to a task executing this transaction. This storage includes task life-time storage—the transaction

work area (TWA) and the EXEC interface block (EIB)—and the storage that CICS obtains on behalf of programs that run under the task.

CVDA values are:

CICSDATAKEY

CICS-key storage is assigned.

USERDATAKEY

User-key storage is assigned.

TASKDATALOC(*cvda*)

returns a CVDA value indicating whether task-lifetime storage for a task executing this transaction should be above or below the 16MB line. Task-lifetime storage includes the EIB and TWA. CVDA values are:

ANY Task-lifetime storage can be above or below the 16MB line.

BELOW

Task-lifetime storage must be below the 16MB line.

| **TCLASS**(*data-area*)

| returns a fullword binary field giving the number of the transaction class to which the transaction belongs, if the task belongs to a numbered class. Zero is returned if the transaction does not belong to any class, and an INVREQ exception condition is raised if the transaction belongs to a class that does not correspond to a numbered class.

| The TCLASS option is retained for compatibility with earlier releases of CICS, where transaction classes were numbered from 0 to 10. In this release, transaction classes have 8-character names, specified by the TRANCLASS value in the definition (see that option in this command).

| A class is numbered only if its name is of the form DFHTCLnn, where “nn” is a number from 00 to 10, and it is this number that is returned by the TCLASS option in this command. (The TRANSACTION definition can contain a TCLASS value as well, to allow the same definition to be installed in a system running under an earlier release, but the TCLASS value is ignored in this release and does not need to correspond to the TRANCLASS value.)

TRACING(*cvda*)

returns a CVDA value indicating the type of tracing to be done for tasks executing this transaction. CVDA values are:

SPECTRACE

Tracing is to be special.

SPRSTRACE

Tracing is suppressed.

STANTRACE

Tracing is to be standard.

+ If this value is other than SPRSTRACE and the task has
+ a principal facility, the tracing value for the task is

+ determined from a combination of the TRACING values for its terminal and the transaction it is executing. In this case, tracing is special if either the terminal or the transaction specifies SPECTRACE, standard if both specify STANTRACE.

+ A TRACING value of STANTRACE is assigned when the transaction is defined. You can specify other values only with a SET TERMINAL command or the CICS-supplied CETR transaction.

| **TRAN**CLASS(*data-area*)

| returns the 8-character name of the transaction class to which this transaction belongs. If the transaction does not belong to any class, the value DFHTCL00 is returned.

TRANSACTION(*data-value*)

specifies the 4-character name of the TRANSACTION definition about which you are inquiring.

TRPROF(*data-area*)

returns the 8-character name of the PROFILE definition used to define attributes associated with the session used for routing, if transaction routing occurs.

TWASIZE(*data-area*)

returns a fullword binary field giving the size, in bytes, of the transaction work area (TWA) for this transaction.

Conditions

| **END**

RESP2 values:

| **2** There are no more resource definitions of this type.

| **IL**LOGIC

RESP2 values:

| **1** You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

INVREQ

RESP2 values:

| **3** The TCLASS option has been specified in this INQUIRE command, and the transaction belongs to a class that is not one of the numbered classes DFHTCL00 through DFHTCL10.

| **10** The PROFILE definition associated with the transaction is not available.

NOTAUTH

RESP2 values:

| **100** The user associated with the issuing task is not authorized to use this command.

INQUIRE TSQUEUE

- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

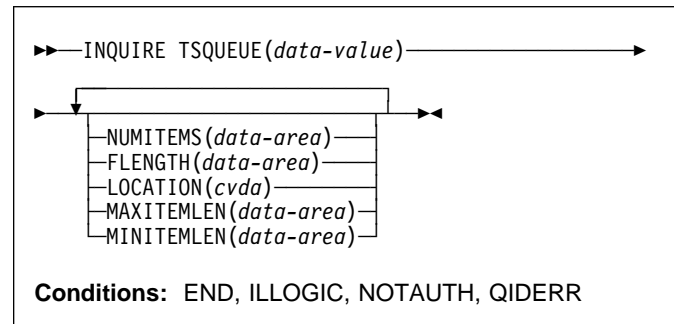
TRANSIDERR

RESP2 values:

- 1 The transaction could not be found.

INQUIRE TSQUEUE

Retrieve information about a temporary storage queue.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE TSQUEUE command returns information about a particular temporary storage queue.

Browsing

You can also browse through all of the temporary storage queues in your system by using the browse options (START, AT, NEXT, and END) on INQUIRE TSQUEUE commands. In browse mode, the definitions are returned in alphabetic order, and you can specify a starting point with the AT option if you wish. If you want to see all the queues with names beginning with a certain string of letters, for example, you can start your browse with an AT value of those letters, padded on the right to 8 characters with nulls (X'00').

In a browse, CICS returns all queues, and you may see queues created by CICS for internal use as well as those created by user applications. In particular, queues with names that start with these characters are CICS queues: `**`, `$$`, `X'FA'` through `X'FF'`, `CEBR` and `DF`.

See “Browsing resource definitions” on page 13 for general information about browsing, syntax, exception conditions, and examples.

Options

NUMITEMS(data-area)

returns a halfword binary field giving the number of items in the temporary storage queue.

FLENGTH(data-area)

returns a fullword binary field giving the total length in bytes of all the items in the temporary storage queue. For information about how CICS calculates the length of items, see the MAXITEMLEN option.

LOCATION(*cvda*)

returns a CVDA value indicating where the temporary storage queue resides. CVDA values are:

AUXILIARY

The temporary storage queue is held in the CICS temporary storage VSAM data set.

MAIN

The temporary storage queue is held in main storage.

MAXITEMLEN(*data-area*)

returns a halfword binary field giving the length in bytes of the largest item in the temporary storage queue.

The length of a queue item is the sum of the length of the user data plus 24 bytes for header information, rounded up. For main storage queues, the length is rounded up to the boundary of the MVS storage subpool used to store it; subpool boundaries are always some multiple of 64.

For auxiliary temporary storage, the length is rounded to the next higher multiple of either 64 or 128 (depending on the control interval size of the temporary storage data set). However, if any item in the queue exceeds the control interval size, -1 is returned for all three lengths associated with the queue (FLENGTH, MAXITEMLEN and MINITEMLEN). (For background information about CI sizes, see the *CICS/ESA System Definition Guide*.)

MINITEMLEN(*data-area*)

returns a halfword binary field giving the length in bytes of the smallest item in the temporary storage queue. For information about how CICS calculates the length of items, see the MAXITEMLEN option.

TSQUEUE(*data-value*)

specifies the 8-character name of the temporary storage queue about which you are inquiring.

- 101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

QIDERR

RESP2 values:

- 1** The temporary storage queue cannot be found.

Conditions**END**

RESP2 values:

- 2** There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1** You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH

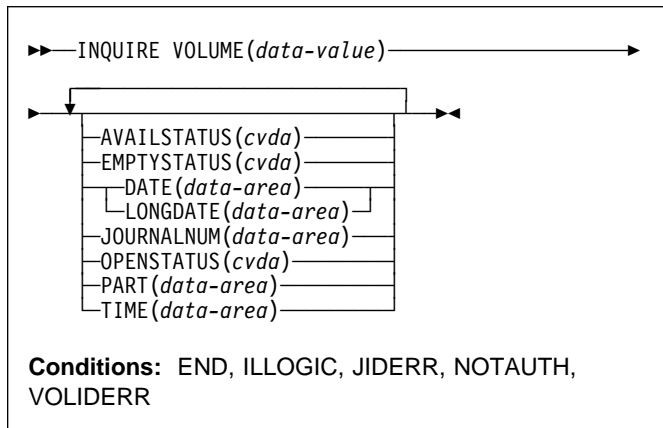
RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

INQUIRE VOLUME

INQUIRE VOLUME

Retrieve information about a journal tape volume.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

The INQUIRE VOLUME command retrieves information about a tape volume that belongs to a CICS journal. It applies only to standard-labeled tapes.

Browsing

You can also browse through all of the volumes in your tape journals by using the browse options (START, NEXT and END) on INQUIRE VOLUME commands, and you can browse selectively by specifying a JOURNALNUM value on your INQUIRE VOLUME NEXT commands.

The JOURNALNUM option both specifies and returns information in a volume browse. Here is how the value specified governs the browse:

- If JOURNALNUM is not specified, or the value is outside the valid range of 1 through 99, the browse continues from the current position (journal and volume); in the first INQUIRE NEXT, this is the first volume of the first tape journal.
- If the JOURNALNUM value is the same as was returned on the previous INQUIRE NEXT, the browse also continues from the current position, giving the next volume in that journal if there is one, or the first volume in the next tape journal if not.
- If the JOURNALNUM value is different from that on the previous INQUIRE NEXT and specifies a tape journal, the first volume in the indicated journal is returned. If the value is in range (1 through 99) but specifies a journal that does not exist or is not a tape journal, the JIDERR condition is raised.

Options

AVAILSTATUS(*cvda*)

returns a CVDA value identifying the availability of the volume. CVDA values are:

FCLOSE

The volume is unavailable because of a failure on close.

FINPUT

The volume is unavailable because of a failure during input.

FOPEN

The volume is unavailable because of a failure on open.

FOUTPUT

The volume is unavailable because of a failure during output.

NOWRITE

The volume cannot be written to.

OK

The volume can be written.

DATE(*data-area*)

returns the date on which the earliest data in the volume was written for the journal to which it belongs. The date is a 5-character value in Julian form, "yyddd". If no data has been written, "00000" is returned.

EMPTYSTATUS(*cvda*)

returns a CVDA value identifying whether the volume is empty or not. CVDA values are:

EMPTY

The volume does not contain data for the journal to which it belongs.

NOTEMPTY

The volume contains data for the journal to which it belongs.

JOURNALNUM(*data-area*)

returns a halfword binary field giving the number of the journal to which the volume about which you are inquiring belongs. A value of 1 identifies the system log.

LONGDATE(*data-area*)

returns the same date as the DATE option, but as a 7-character string that contains the full four-digit year, in the form "yyyyddd".

OPENSTATUS(*cvda*)

returns a CVDA value indicating whether the volume is open or closed. CVDA values are:

CLOSED

The volume is closed.

OPEN

The volume is open.

OPENING

The volume is in the process of opening.

PART(*data-area*)

returns a fullword binary field giving the sequence number of this volume in a series of volumes.

TIME(*data-area*)

returns a 6-character value, in the format "hhmmss", giving the time at which the earliest data in the volume was written for the journal to which it belongs. If no data has been written, "000000" is returned.

VOLUME(*data-value*)

specifies the 6-character volume serial number of the volume about which you are inquiring.

Conditions**END**

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

JIDERR

RESP2 values:

- 3 The journal specified with the NEXT option cannot be found.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.
- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

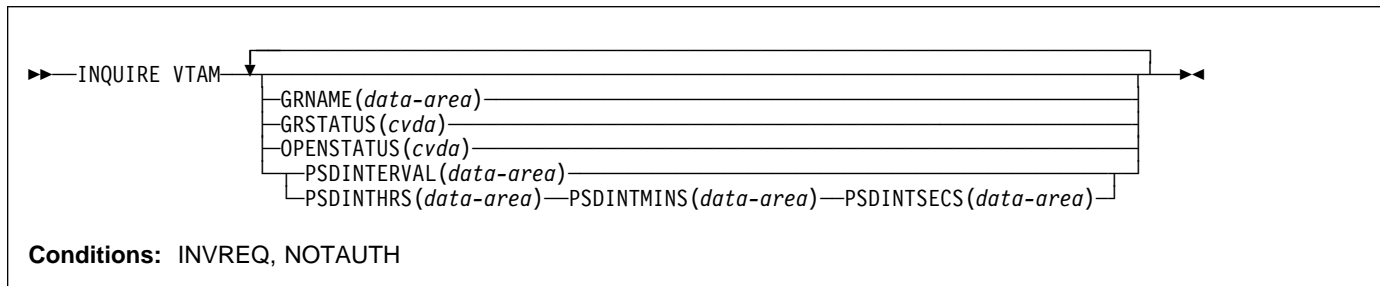
VOLIDERR

RESP2 values:

- 1 The volume cannot be found.

INQUIRE VTAM

Retrieve information about the connection between CICS and VTAM.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The INQUIRE VTAM command returns information about type and state of the connection between VTAM and your CICS system.

Options

GRNAME(*data-area*)

returns the 8-character generic resource group name under which this CICS region requests registration to VTAM, if it is using the generic resources facility of VTAM. Blanks are returned if the system was initialized without a request for registration.

GRSTATUS(*cvda*)

returns a CVDA value indicating the status of generic resource registration. All of the values except NOTAPPLIC indicate that CICS has been initialized to use the generic resource function (that is, that a nonblank GRNAME value was specified). CVDA values are:

DEREGERROR

Deregistration was attempted but was unsuccessful, and there has been no attempt to re-register.

DEREGISTERED

Deregistration was successfully accomplished.

NOTAPPLIC

CICS is not using the generic resource feature; GRNAME is not set or is set to blanks.

REGERROR

Registration was attempted but was unsuccessful, and there has been no attempt to deregister.

REGISTERED

Registration was successful and there has been no attempt to deregister.

UNAVAILABLE

VTAM does not support the generic resource function.

UNREGISTERED

CICS is using the generic resource function but no attempt, as yet, has been made to register.

OPENSTATUS(*cvda*)

returns a CVDA value indicating the status of the connection between CICS and VTAM. CVDA values are:

CLOSED

The connection between CICS and VTAM has not yet been established or has been terminated.

+ CLOSEFAILED

+ The connection is open but is not usable because a previous request to close the connection failed.
+ You should retry the close request.

CLOSING

The connection between CICS and VTAM is in the process of closing.

FORCECLOSING

The connection between CICS and VTAM is in the process of closing following a SET VTAM FORCECLOSE command.

IMMCLOSING

The connection between CICS and VTAM is in the process of closing following a SET VTAM IMMCLOSE command.

OPEN

There is a connection between CICS and VTAM.

PSDINTERVAL(*data-area*)

returns the persistent session delay (PSD) interval, which is the length of time that sessions are held in recovery-pending state after a CICS failure. (See the PSDINT system initialization parameter in the *CICS/ESA System Definition Guide* for more information about this option.) There are two formats for the PSD interval:

- A composite (packed decimal format **0hhmmss+**, 4 bytes long), which you obtain by using the PSDINTERVAL option.

- | • Separate hours, minutes, and seconds, which you
 - | obtain by specifying the PSDINTHRS,
 - | PSDINTMINS, and PSDINTSECS options.
- + (A value of zero means that sessions will not be held
- + after a failure, and may indicate that the VTAM in use is
- + not at the level that supports persistent sessions.)

| **PSDINTHRS**(*data-area*)

| returns the hours component of the PSD interval, in

| fullword binary form (see the PSDINTERVAL option).

| **PSDINTMINS**(*data-area*)

| returns the minutes component of the PSD interval, in

| fullword binary form (see the PSDINTERVAL option).

| **PSDINTSECS**(*data-area*)

| returns the seconds component of the PSD interval, in

| fullword binary form (see the PSDINTERVAL option).

Conditions

INVREQ

RESP2 values:

- | **1** VTAM is not present in the system.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

PERFORM DELETSHIPED

- + Delete inactive shipped terminal definitions.

▶▶—PERFORM DELETSHIPED—◀◀

Conditions: NOTAUTH

Description

- + The PERFORM DELETSHIPED command causes
- + immediate invocation of the CICS mechanism for deleting
- + inactive shipped terminal definitions. It does **not** reset the
- + interval at which this mechanism is normally invoked. That
- + is, it does not affect the time remaining until the next
- + automatic invocation.

- + A shipped definition is inactive if the terminal has not been
- + used locally for a specified period of time and no task is
- + waiting to be attached which requires the terminal. You can
- + determine the length of time a shipped terminal must remain
- + unused to be eligible for deletion and the interval at which
- + CICS checks for such terminals with the INQUIRE
- + DELETSHIPED command, and you can set these values
- + with the SET DELETSHIPED command. For more
- + information about shipped definitions, see the *CICS/ESA*
- + *Intercommunication Guide* and *CICS/ESA Resource*
- + *Definition Guide* manuals.

Conditions

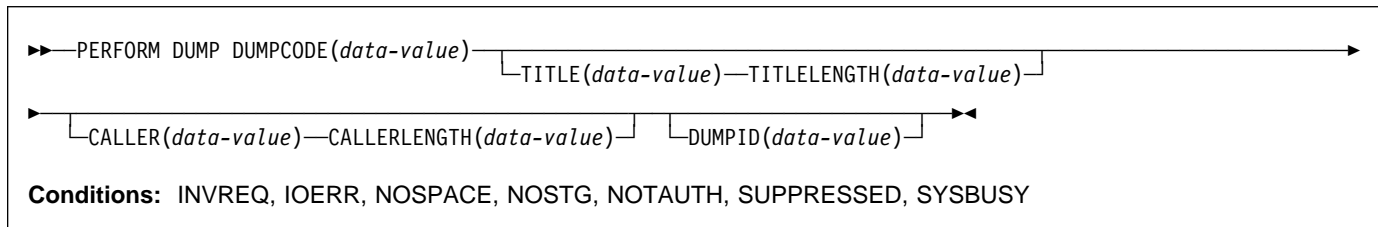
NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

PERFORM DUMP

+ Request a system dump of CICS.



Description

- + The PERFORM DUMP command requests a system dump (an MVS SDUMP) of the CICS region in which it is issued.
- + The system dump table entry for the dump code specified in the DUMPCODE option determines the processing that occurs on a PERFORM DUMP command: whether a dump is taken, whether the request is propagated for related CICS regions in a sysplex environment, and whether shutdown occurs. If there is no entry for the dump code you specify, CICS creates a temporary one using default values. See the INQUIRE SYSDUMPCODE command for more information about this process and the *CICS/ESA Problem Determination Guide* for general information about the system dump table.
- + While an MVS SDUMP is being taken, all other CICS activity ceases. The program issuing the command does not regain control until the dump is complete, and then only if the dump does not cause CICS to shut down.

Options

CALLER(data-value)

- + specifies the text that appears after 'CALLER' in the summary of dump domain information at the top of the dump. This text can be up to 8 characters long. It is intended to identify the source of the request for the dump, but is not restricted to that purpose.

CALLERLENGTH(data-value)

- + specifies, as a fullword binary value, the number of characters in the CALLER text.

DUMPCODE(data-value)

- + specifies the 8-character dump code for this dump request, which determines the system dump table entry used in processing it.
- + The code can be either CICS-defined or user-defined. Most CICS codes are a CICS message identifier with the initial 'DFH' removed, but there are a few additional ones. The *CICS/ESA Messages and Codes* manual lists all CICS messages and also the additional codes (under "System dump codes").
- + User-defined codes can be any character string that does not contain leading or imbedded blanks.

- + CICS provides system dump table entries for some CICS-defined codes and builds them as needed for others. The installation can provide entries for user-defined codes, or CICS will build temporary entries, as explained above.

DUMPID(data-value)

- + specifies a 9-character identifier for this particular dump.
- + This value is intended to distinguish this dump uniquely from any others, but it is not restricted to this use.

TITLE(data-value)

- + specifies the text that is printed as a title in the summary of dump domain information at the top of the dump. It can be up to 80 characters long.

TITLELENGTH(data-value)

- + specifies, as a fullword binary value, the number of characters in the TITLE text.

Conditions

INVREQ

RESP2 values:

- 6** TITLELENGTH is greater than 80 bytes.
- 7** CALLERLENGTH is greater than 8 bytes.
- + **13** The DUMPCODE contains leading or imbedded blanks.

IOERR

RESP2 values:

- + **9** CICS is not authorized by MVS to take dumps.
- 10** An error occurred during system dumping.
- + **12** MVS cannot process the dump because there is no dump data set or because it is full.
- + **13** An error occurred in the CICS routine that issues MVS SDUMP requests.

NOSPACE

RESP2 values:

- + **4** The dump is incomplete due to lack of dump data-set space.

NOSTG

RESP2 values:

- + **5** CICS cannot complete the dump because of
- + insufficient storage.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

SUPPRESSED

RESP2 values:

- + **1** The dump was not taken because the number of
- + dumps with this dump code exceeds the maximum
- + for the code.
- + **2** The dump was not taken because the system
- + dump table entry for this code indicates no system
- + dump.
- + **3** The dump was not taken because it was
- + suppressed by a user exit program.
- + **8** The dump was not taken because system dumps
- + are suppressed globally.

SYSBUSY

RESP2 values:

- + **11** The MVS dump routine is busy. Retry the
- + command.

PERFORM RESETTIME

Reset date and time.

▶▶—PERFORM RESETTIME—◀◀

Conditions: INVREQ, NOTAUTH**Description**

- + The PERFORM RESETTIME command resets the CICS date
- + and time from the MVS system date and time.

Conditions**INVREQ**

RESP2 values:

- 1** There is no clock in the system.

NOTAUTH

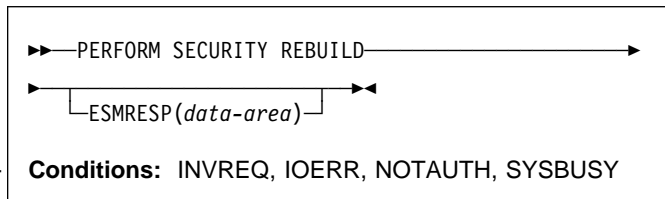
RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

PERFORM SECURITY REBUILD

PERFORM SECURITY REBUILD

+ Refresh security information.



+ **Conditions:** INVREQ, IOERR, NOTAUTH, SYSBUSY

Description

+ The PERFORM SECURITY REBUILD command is a request
+ for CICS security information to be refreshed from its
+ external security manager (ESM) source, so that it reflects
+ any updates made since the information was last retrieved.

+ When the ESM is RACF at a release level below Version
+ 2.1, PERFORM SECURITY REBUILD causes the in-memory
+ copies of CICS resource profiles to be rebuilt, so that they
+ are entirely current. If a large number of RACF profiles must
+ be rebuilt, the process can take several minutes. During this
+ time, any CICS task that requires a security check in order to
+ continue must wait until the rebuild is completed, and the
+ performance of other CICS tasks can be severely affected.

+ With RACF Version 2.1, the refresh process is automatic;
+ PERFORM SECURITY REBUILD is not required, and has no
+ effect if issued.

+ If your CICS uses another ESM, the effect of this command
+ depends on the particular ESM.

Options

ESMRESP(*data-area*)

+ returns a fullword binary field giving the response code
+ from the external security manager. This value is also
+ returned in the RESP2 field of the response code. If an
+ exception condition prevents CICS from invoking the
+ ESM, the ESMRESP value is left unchanged.

+ When the ESM is RACF and the command completes
+ normally (a RESP code value of NORMAL), a RESP2
+ code of 0 indicates that the profiles have been
+ refreshed. A RESP2 code of 4 means that no action
+ was required, because RACF was at Version 2.1 or
+ greater, in which profiles are refreshed automatically.

Conditions

INVREQ

RESP2 values:

+ **1** No ESM is installed, or the ESM is inactive.
+ **5** The ESM is temporarily inactive and cannot
+ perform the action requested.

IOERR

RESP2 values:

+ **3** Error returned from ESM. The return code is in
ESMRESP, if the option was used.

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not
authorized to use this command.

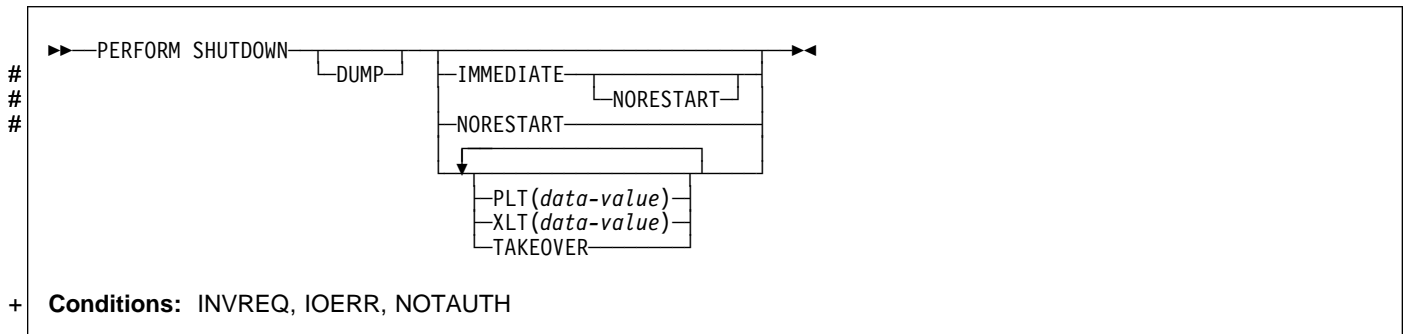
SYSBUSY

RESP2 values:

| **3** A security rebuild is currently in progress.

PERFORM SHUTDOWN

Shut down the CICS system.



Description

The PERFORM SHUTDOWN command shuts down the CICS system. The shutdown can be either normal (controlled) or immediate. Control does not return to the program issuing the command, unless an exception condition occurs.

In processing this command, CICS invokes the programs in the shutdown program list table (PLT) as part of the task that issued the command. If any program in the list requires a terminal (that is, uses the principal facility), you should not issue the command in a task that does not have one, because the task will abend on the first attempt to use the non-existent terminal. Shutdown will proceed, but the task will be backed out to its most recent SYNCPOINT, and the remaining programs in the list will not be executed.

The *CICS/ESA Customization Guide* contains more information about PLTs and steps in the shutdown process.

Options

DUMP

specifies that an MVS SDUMP of the CICS region should be taken as part of the shutdown process. In a sysplex environment, dumps of related regions also will be taken, if the system dump table entry for the dump code SHUTDOWN, which governs this dump, specifies them.

IMMEDIATE

specifies that CICS is to shut down immediately, terminating all active tasks and VTAM sessions abnormally. If IMMEDIATE is not specified, CICS shuts down normally, allowing these tasks to complete and quiescing the sessions.

NORESTART

specifies that this CICS region should not be restarted (by MVS automatic restart manager) after the CICS region has completed shutting down.

This option applies to immediate shutdowns only. If you specify NORESTART without IMMEDIATE, CICS

performs an immediate shutdown as if you specified the command as PERFORM SHUTDOWN IMMEDIATE NORESTART.

PLT(*data-value*)

specifies the 2-character suffix that identifies the PLT for this shutdown. (The table is a load module named DFHPLT followed by this suffix.)

The value "NO" means that no PLT programs are run. If you do not supply a PLT value, the value specified by the PLTSD system initialization parameter, if any, is used. This option applies only to a normal shutdown; the PLT is not run in an immediate shutdown.

TAKEOVER

specifies that this CICS system is to be shut down normally, and then the alternate CICS system is to take over. This option is valid only when the system initialization parameter XRF=YES was specified for CICS startup.

XLT(*data-value*)

specifies the 2-character suffix that identifies the transaction list table (XLT) to be used for this shutdown. (The table is a load module named DFHXLT followed by this suffix.)

This table lists the transactions that can be initiated by unsolicited terminal input during the first quiesce stage of a normal shutdown. No other transactions can be initiated from a terminal during shutdown, except for CEMT, CESF, and a small number of other CICS-supplied transactions related to terminals.

This option is meaningful only when IMMEDIATE is not present; no new transactions are accepted during an immediate shutdown. A suffix of "NO" means that no transactions besides those cited above are allowed. If you do not supply an XLT value, the value specified by the XLT system initialization parameter, if any, is used.

Conditions

PERFORM STATISTICS RECORD

INVREQ

RESP2 values:

- + 1 A normal shutdown was requested when shutdown was already in progress.
- + 2 The XLT cannot be found.
- + 3 The PLT cannot be found.
- + 4 XRF is not in effect.

IOERR

RESP2 values:

- + 13 The request for an MVS SDUMP failed.

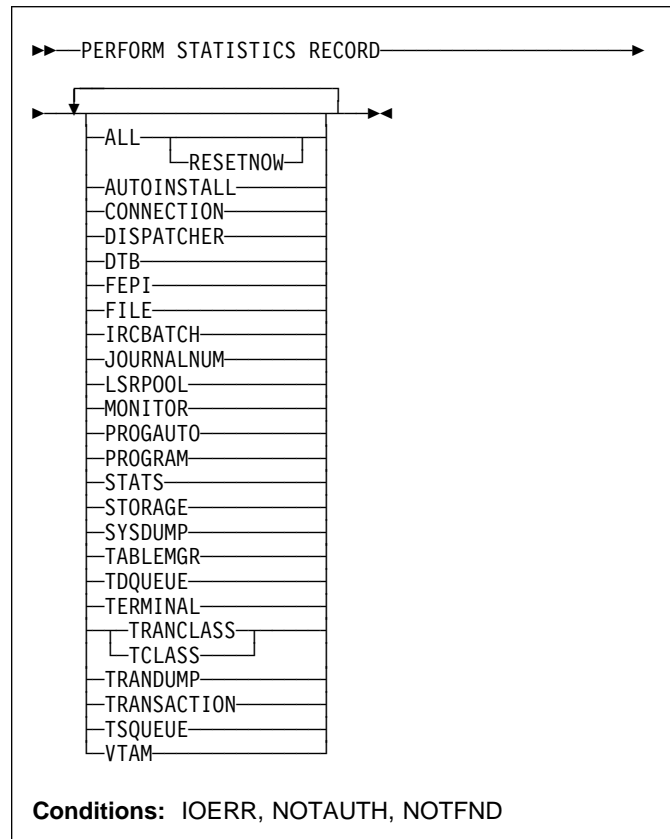
NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

PERFORM STATISTICS RECORD

Record statistics immediately.



Description

The PERFORM STATISTICS RECORD command causes + current statistics for the resource types and system functions + that you specify to be recorded (written out to the SMF data + set). Recording occurs immediately, and is not governed by + the system options that control the recording of these + statistics at intervals. (See the discussion of **interval** + **statistics** in the SET STATISTICS command on page 172.)

+ Execution of this command does not affect interval or + end-of-day statistics either, except when you specify + RESETNOW, because the counts are not reset unless + RESETNOW is specified.

+ You can specify as many types of statistics as you wish, or + you can request all (the ALL option). For each type you + request, CICS provides all of the information available (the + information that is recorded in interval statistics). For system + services, such as dispatch and dynamic transaction backout, + CICS keeps summary (global) statistics. For resource types, + CICS keeps specific statistics for each installed resource of + the type in question, and for some resource types, CICS + keeps global counts as well.

- + The *CICS/ESA Performance Guide* contains details about
- + CICS statistics.

+ Options

+ ALL

- + records statistics for all resource types and system services. This is the same information that is recorded for interval statistics, and includes counts from the user domain, which are not otherwise available with this command.
- + In addition, you can reset the counts when you use this option (see the RESETNOW option).

+ AUTOINSTALL

- + records global statistics on the automatic installation of terminal definitions.

+ CONNECTION

- + records specific statistics for all connections installed in your system.

+ DISPATCHER

- + records global statistics on the dispatch function, including task counts and concurrency levels and limits.

+ DTB

- + records global statistics on the dynamic transaction backout function.

+ FEPI

- + records global statistics on the front-end programming interface (FEPI) and specific statistics on FEPI connections, targets and pools.

+ FILE

- + records specific statistics for all files and DL/I data bases installed in your system.

+ IRCBATCH

- + records global statistics on the IRC batch function (shared use of DL/I data bases through interregion communication).

+ JOURNALNUM

- + records specific statistics for all journals installed in your system.

+ LSRPOOL

- + records specific statistics on all VSAM LSR pools defined in your system, including statistics on the files within the pool additional to the statistics produced by the FILE option.

+ MONITOR

- + records global statistics on the monitor function of CICS.

+ PROGAUTO

- + records global statistics on automatic installation of program definitions.

+ PROGRAM

- + records global and specific statistics for all programs installed in your system.

+ RESETNOW

- + resets all statistics to initial values after recording. You can use this option only in conjunction with the ALL option. The definition of the initial value depends on the statistic being kept; see the *CICS/ESA Performance Guide* for details.

+ STATS

- + records global statistics about the statistics-gathering function of CICS.

+ STORAGE

- + records global statistics for all CICS dynamic storage subpool areas, and specific statistics by subpool.

+ SYSDUMP

- + records global statistics on system dumps and specific statistics for each dump code in the system dump code table.

+ TABLEMGR

- + records global statistics on the CICS table manager.

+ TCLASS

- + records specific statistics for every transaction class defined in your system. This option has the same effect as TRANCLASS and is retained for compatibility with older versions of CICS only; use TRANCLASS instead where possible.

+ TDQUEUE

- + records global statistics for transient data and specific statistics for each queue defined in your system.

+ TERMINAL

- + records specific statistics for each terminal and session installed in your system.

+ TRANCLASS

- + records specific statistics for every transaction class defined in your system.

+ TRANDUMP

- + records global statistics on transaction dumps and specific statistics for each dump code in the transaction dump table.

+ TRANSACTION

- + records global statistics on transactions and specific statistics for each transaction installed in the system.

+ TSQUEUE

- + records global statistics on temporary storage.

+ VTAM

- + records global VTAM statistics for your system.

PERFORM STATISTICS RECORD

Conditions

IOERR

RESP2 values:

- + **n** Statistics for at least one of the options chosen
- + were not available; usually the reason for this error
- + is corruption of the memory in which they are
- + accumulated. (See note below.)

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

NOTFND

RESP2 values:

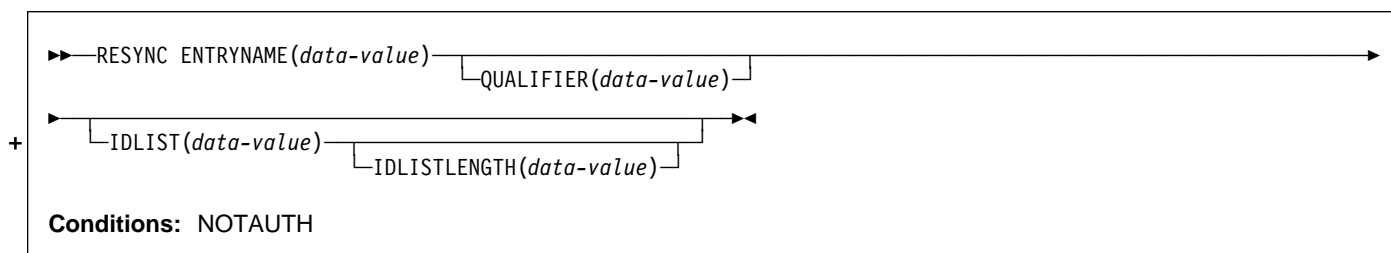
- + **n** Statistics for at least one of the options chosen
- + were not available because CICS was initialized
- + without support for the function. (See note below.)

- + **Note:** When statistics of a requested type are unavailable,
- + CICS raises the IOERR or NOTFND exception condition, as
- + appropriate, but continues through the remaining types,
- + recording as much information as available. The RESP2
- + value "n" identifies the last type to fail in this way, as follows:

	n	Resource type
	1	AUTOINSTALL
	2	CONNECTION
+	3	DISPATCHER
	5	DTB
	6	FILE
	7	IRCBATCH
+	8	JOURNALNUM
	10	LSRPOOL
	11	MONITOR
	12	PROGRAM
	13	STATS
	14	STORAGE
	15	SYSDUMP
+	16	TABLEMGR
	18	TCLASS, TRANCLASS
	19	TDQUEUE
	20	TERMINAL
	21	TRANDUMP
	22	TRANSACTION
	23	TSQUEUE
	24	VTAM
	25	FEPI
+	26	PROGAUTO

RESYNC ENTRYNAME

Determine the disposition of “in doubt” units of work.



Description

The RESYNC command allows a non-CICS resource manager to determine whether units of work about which it is “in doubt” were committed or backed out.

A resource manager can be in doubt about a unit of work if it has been invoked for the first phase of syncpoint, but not for the second, when it is told whether to commit or roll back. A failure of either the resource manager or CICS between Phase 1 and Phase 2 leaves the resource manager in doubt about that unit of work.

CICS saves or reconstructs the disposition of any such unit of work until a RESYNC command or a cold start. CICS also saves the disposition of any unit of work about which the resource manager replies “remember” to the second-phase syncpoint invocation, so that if the resource manager cannot commit or roll back as directed, it can request the disposition later for recovery.

To use the saved disposition information, the resource manager must have a record of which units of work are in doubt or “remembered.” It can then issue a RESYNC command with a list of these units of work, either in its task-related user exit or an associated administrative transaction.

In response, CICS creates a task, CRSY, for each in-doubt in the list. The CRSY task invokes the task-related user exit once on behalf of its particular unit of work. This invocation is identified to the exit as a phase 2 syncpoint request and as such indicates whether the unit of work was committed or rolled back. The exit can then relay this information in the form the resource manager requires.

At this time CICS also **discards** all other disposition information it has kept for the resource manager. Therefore, the resource manager must resynchronize all in-doubts at once. A resource manager is identified by the name of its task-related user exit and, optionally, a qualifier to this name. Use of a qualifier allows multiple instances of the same resource manager to resynchronize independently.

Control is returned to the program that issued the RESYNC command as soon as the CRSY tasks have been scheduled. They run asynchronously, in parallel, according to normal

CICS dispatch rules. Consequently, the exit should be enabled, started and initialized to the point where it can process these invocations before the RESYNC command.

If the exit is not available, a CRSY task will save the disposition of its unit of work, but since this occurs later in time, no exceptional condition occurs on the RESYNC. See the *CICS/ESA Customization Guide* for full details about resynchronization invocations of task-related user exits.

Options

ENTRYNAME(*data-value*)

specifies the 8-character name of the task-related user exit for the resource manager. This is the ENTRYNAME value of the ENABLE command that established the exit, or, if ENTRYNAME was omitted, the PROGRAM value.

+ IDLIST(*data-value*)

- + specifies the list of units of work to be resynchronized.
- + Each entry in the list is the *address* of the 8-byte identifier of an in-doubt unit of work. The end of the list may be indicated by the high-order bit turned on, or IDLISTLENGTH may be used.
- + Units of work are identified by the UEPURID value passed to the task-related user exit.

Note: IDLIST is optional, but if you omit it, CICS discards all of the saved disposition information for the resource manager.

IDLISTLENGTH(*data-value*)

specifies a halfword binary value indicating the length (in bytes, counting 4 bytes per in-doubt) of the address-list.

QUALIFIER(*data-value*)

- + specifies an 8-character qualifier to the ENTRYNAME value, which identifies the particular instance of the resource manager to which the RESYNC command applies. The qualifier is optional; it is intended for systems where more than one copy of a resource manager can be in use.
- + When it is in use, this value is assigned to a unit of work by the task-related user exit at the time the unit of work takes place, via the UEPRMQUA value in the user exit parameter list. If the RESYNC command specifies a qualifier, CICS uses only disposition information saved

SET AUTOINSTALL

with the same QUALIFIER and ENTRYNAME values. Similarly, it discards saved dispositions only if they have the same two values and were not included in the IDLIST.

Conditions

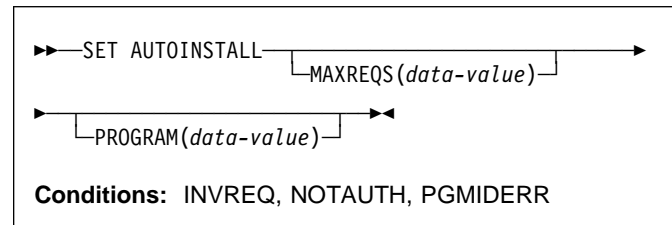
NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

SET AUTOINSTALL

Change autoinstall values.



Description

The SET AUTOINSTALL command lets you change some of the values that control the automatic installation of VTAM terminals (autoinstall) in your CICS system.

Options

MAXREQS(*data-value*)

- + specifies the largest number of autoinstall requests that can be processed concurrently, as a fullword binary value. The value must be in the range 0–999.
- + **Note:** MAXREQS does not limit the total number of terminals that can be installed automatically, only the arrival rate. However, you can prevent automatic installation of any additional terminals by setting MAXREQS to 0. Terminals already autoinstalled are not affected, but if they log off, they cannot log on again while MAXREQS is 0.

PROGRAM(*data-value*)

- + specifies the 8-character name of the program to be used in the autoinstall process for terminals. You can specify either an installation-specific program or the CICS-supplied default, DFHZATDX.
- + **Note:** This program and any programs it invokes must be installed before they can be used in the terminal autoinstall process. You can do this either with explicit PROGRAM definitions or by autoinstall when some other autoinstall program is in force. Otherwise, the terminal autoinstall process fails when it is next used.

Conditions

INVREQ

RESP2 values:

- 1** VTAM is not in use in this system.
- 2** The MAXREQS value is not in the range 0–999.
- + **4** One of the modules invoked by DFHZATDX (DFHZATA and DFHZATD) cannot be found.
- +

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

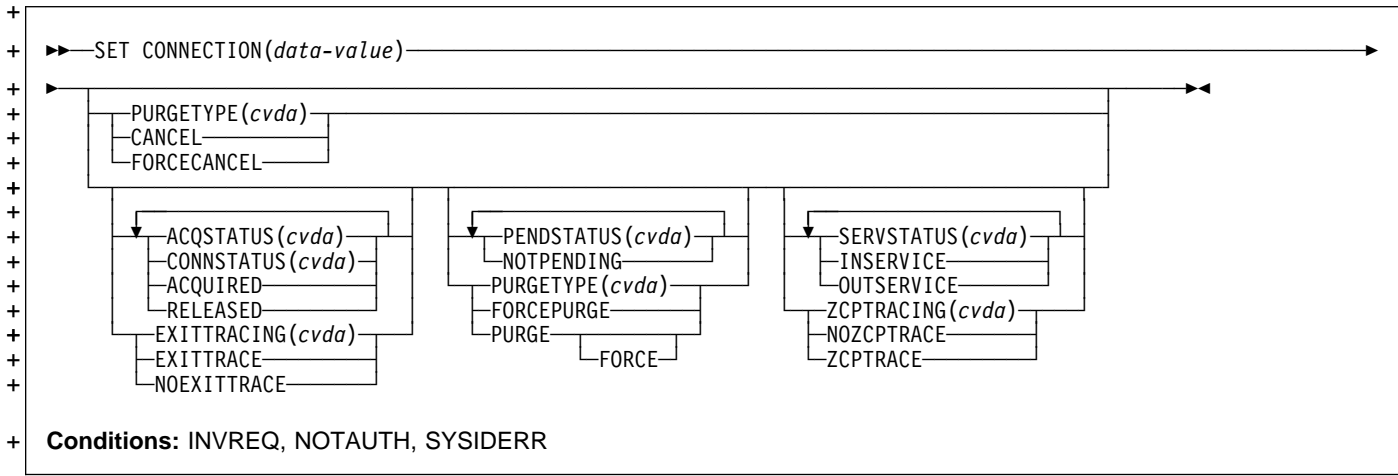
PGMIDERR

RESP2 values:

+ **3** The program name cannot be found.

SET CONNECTION

+ Change the definition or status of a connection or cancel work associated with it.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

+ CICS/ESA *Intercommunication Guide* for more information
 + about ISC and MRO connections, and the CICS/ESA
 + *External CICS Interface* for more information about EXCI.)

Description

The SET CONNECTION command allows you to change both the definition and the current status of a connection between your local CICS region and another CICS region or another system. A CONNECTION definition is sometimes known as a "system entry."

+ For some options, CICS returns control to the task that issued the SET CONNECTION command after the request is scheduled but before it is completed. You can test for completion, if you need to know, with an INQUIRE CONNECTION command, or, in the case of an APPC connection, with INQUIRE MODENAME in browse mode.

There are two main types of system connection:

- Multiregion operation (MRO), which uses CICS interregion communication (IRC) to establish a connection between two MRO partners. An MRO connection can exist between two CICS regions on the same MVS image or within the same MVS/ESA sysplex.
- Intersystem communication (ISC) connections, between CICS and any other system which supports VTAM APPC or LUTYPE6.1 communications. For example, ISC connections can exist between CICS regions running in different MVS/ESA sysplexes or on different operating system platforms, between CICS and any APPC device, and between CICS and IMS.

+ SET CONNECTION can be used for all of these connections, although not all options apply to all types. Exceptions are noted in the option descriptions that follow. In addition, you cannot use SET CONNECTION for either a remote connection or a connection defined indirectly. (See the

Options

ACQSTATUS(cvda) (APPC only)

+ specifies the same information as the CONNSTATUS option and is retained only for compatibility. You should use CONNSTATUS in new applications.

CONNECTION(data-value)

+ specifies the 4-character name of the connection that you are modifying (that is, the name by which the connected region is known to your CICS system, from its CONNECTION definition).

CONNSTATUS(cvda) (APPC only)

+ specifies whether or not CICS is to be in session with the remote system represented by this connection (that is, whether the connection is to be **acquired** in the VTAM sense).

+ CONNSTATUS applies only to APPC connections in a SET CONNECTION command. For an MRO connection, the CONNSTATUS value is determined by the SERVSTATUS value, and you cannot set it directly, even though you can inquire about it in an INQUIRE CONNECTION command. CONNSTATUS does not apply at all to EXCI and or LUTYPE6.1 connections.

CVDA values are:

ACQUIRED

+ CICS is to be in session, and is to establish a
 + connection if none exists. If the SERVSTATUS
 + value of the connection is OUTSERVICE, you
 + must specify INSERVICE as well as ACQUIRED,
 + because CICS will not acquire an APPC
 + connection that is out of service. In addition, you
 + cannot specify this value if the current
 + CONNSTATUS value is FREEING (see the
 + CONNSTATUS option in the INQUIRE
 + CONNECTION command).

RELEASED

+ CICS is not to be in session, and is to terminate
 + the connection if one exists. CICS delays
 + releasing the connection until any tasks currently
 + using a session on the connection terminate.
 + Consequently, you must specify PURGE or
 + FORCEPURGE as well as RELEASED if sessions
 + are in use and you want the release to occur
 + immediately.

+ **Note:** CICS uses a task that executes LU Services
 + Manager transaction CLS1 to acquire or release
 + sessions on an APPC parallel-session connection. Data
 + is passed to the task in a temporary storage queue
 + whose name begins with the default prefix of DF. If your
 + system defines queues named starting with DF as
 + recoverable, CICS cannot initiate this task until a
 + subsequent commit on the part of the task that changed
 + the CONNSTATUS value (either a SYNCPOINT
 + command or an implicit syncpoint).

EXITTRACING(*cvda*) (VTAM only)

+ specifies whether sessions on this connection are to be
 + traced when CICS is tracing terminal-specific invocations
 + of its VTAM exits (see the TCEXITSTATUS option of the
 + SET TRACEFLAG command). CVDA values are:

EXITTRACE

+ Sessions are to be traced.

NOEXITTRACE

+ Sessions are not to be traced.

PENDSTATUS(*cvda*) (APPC only)

+ specifies that the resynchronization process that
 + normally occurs when a connection is acquired is to be
 + overridden. The CVDA value to request override is
 + NOTPENDING. It causes CICS to:

- + • Commit unilaterally any suspended recoverable
 + changes associated with units of work that required
 + synchronization with the remote system.
- + • Erase its memory of the remote system's log name.
 + This action causes CICS to specify the VTAM option
 + COLD on the next exchange of log names (XLN)
 + with the remote system, which occurs when the
 + connection is re-established. COLD tells the remote
 + system not to send resynchronization information.
 + For this reason, you cannot specify NOTPENDING

+ after a connection has progressed beyond the
 + exchange of log names.

+ XLN and resynchronization are described in the
 + *CICS/ESA Intercommunication Guide* and the *Systems
 + Network Architecture—LU6.2 Reference: Peer Protocols*
 + manual.

PURGETYPE(*cvda*)

+ specifies that work occurring on or scheduled for
 + sessions on this connection is to be canceled.

+ The PURGE and FORCEPURGE values allow you to
 + purge (terminate abnormally) the tasks *currently* using a
 + session on this connection as principal or alternate
 + facility.

+ For such tasks, these options have the same effect in a
 + SET CONNECTION command as they do in a SET
 + TASK or SET TERMINAL command. However, since
 + there may be multiple tasks, deferred purges and tasks
 + exempted because of purgeability status are not
 + reported as exceptions. See page 179 for details on
 + how these options work and precautions about using
 + them.

+ PURGE and FORCEPURGE apply only to APPC and
 + LUTYPE6.1 connections in this command, but you can
 + use them for an individual MRO session in a SET
 + TERMINAL command.

+ The CANCEL and FORCECANCEL options cancel the
 + *scheduled* work represented by the set of automatic
 + initiate descriptors (AIDs) for the connection. Each AID
 + represents a request for a task which is to execute with
 + a session on this connection as its principal facility.

+ These options have the same effect in this command as
 + CANCEL in SET TERMINAL, and you can use them for
 + any type of connection. See page 186 for details. They
 + must be used alone, with no other option specified.

+ The difference between CANCEL and FORCECANCEL
 + is that AIDs for certain CICS-supplied system
 + transactions are exempted from a CANCEL request,
 + whereas FORCECANCEL applies to all. Canceling the
 + exempted AIDs can lead to unpredictable results and,
 + under some circumstances, can cause CICS to
 + terminate abnormally. Consequently, you should use
 + FORCECANCEL only in exceptional situations. Table 3
 + on page 152 lists AIDs that require FORCECANCEL.

SET CONNECTION

Table 3. Transactions for which AIDs require FORCECANCEL	
Remote delete AIDs	CRMD
Remote scheduler AIDs	CRSR
LU6.2 service manager 1 AIDs	CLS1
LU6.2 service manager 2 AIDs	CLS2
LU6.2 service manager 3 AIDs	CLS3
Remote schedule PURGE AIDs	CRSQ
Resource manager resync AIDs	CRSY
Autoinstall terminal delete AIDs	CATD
Restart terminal delete AIDs	CATR

You can determine whether AIDs were canceled by a SET CONNECTION command by inspecting the RESP2 value returned. It will be 58 if any were canceled and 59 if not.

CVDA values are:

CANCEL

AIDs for this connection are to be canceled, except those for the transactions in Table 3.

FORCECANCEL

All AIDs for this connection are to be canceled.

FORCEPURGE

Tasks using this connection are to be terminated as soon as consistent with system integrity and without regard to data integrity.

PURGE

Tasks using this connection are to be terminated as soon as both system and data integrity can be maintained.

PURGE FORCE has been replaced by FORCEPURGE and is retained only for compatibility. You should use FORCEPURGE in new applications.

Note: When a request in one CICS system requires a connection owned by another CICS, both the originating CICS and the CICS that owns the connection maintain an AID to represent the request. If you cancel any such AIDs by issuing a SET CONNECTION CANCEL in the CICS that owns the connection, that CICS notifies the originating CICS systems to cancel their corresponding AIDs. The originating CICS systems write message DFHTF0100 to their CSMT transient data destinations to indicate how many AIDs were canceled for this reason and how many remain for the connection.

SERVSTATUS(*cvda*)

specifies whether the connection is to be available for use. It corresponds to the INSERVICE option in the CONNECTION definition. CVDA values are:

INSERVICE

The connection is to be available.

OUTSERVICE

The connection is not to be available.

The processing that occurs when a connection goes from OUTSERVICE to INSERVICE status, and the status of the individual sessions on the connection, vary with the type of connection, as follows:

- For a parallel-session APPC connection, the LU Services Manager sessions are placed in service, enabling the connection to be acquired, but other sessions on the connection remain out of service until the connection is acquired. For a single-session APPC link, the session itself is marked in service. No attempt is made to establish communication in either case unless ACQUIRED is also specified.
- For a CICS-to-CICS MRO connection, all sessions on the connection are placed in service. If interregion communication is started, CICS attempts to establish communication with the remote system. Success requires that IRC be open on the remote region as well, and that its definition of the connection also be in INSERVICE status. When all of these conditions are met, the connection becomes ACQUIRED as well as INSERVICE (see the CONNSTATUS option).
- For an EXCI connection, all receive sessions (pipes) are placed in service, available to a client program that establishes communication.
- For an LUTYPE6.1 connection, all sessions are placed in service, but communication is not attempted.

Similarly, the processing that occurs when a connection goes from INSERVICE to OUTSERVICE status, and the status of individual sessions, vary with the type of connection.

- For a parallel-session APPC connection, OUTSERVICE causes the LU Services Manager sessions to be placed out of service, so that the connection cannot be reacquired until the connection is put in service again. For single-session APPC, the session is marked out of service, with the same effect. (The connection must be released before it can be placed out of service; you can request RELEASED in the same command if necessary.)
- For other types of connection, activity is quiesced. Tasks using a session on the connection are allowed to complete, but no additional tasks are allowed to use the connection. (Tasks running under the execution diagnostic facility (EDF) complete, but without EDF.) If the connection is LUTYPE6.1, you can specify PURGE or FORCEPURGE to terminate any such tasks immediately. Once the connection is quiesced, the sessions are placed out of service. CICS then

- + releases the connection if it is currently in
- + ACQUIRED status.

ZCPTRACING(*cvda*) (VTAM only)

- + specifies whether sessions on this connection are to be
- + traced when CICS tracing for VTAM sessions is turned
- + on (this is the tracing controlled by the TC option in the
- + SET TRACETYPE command). CVDA values are:

NOZCPTRACE

- + Sessions are not to be traced.

ZCPTRACE

- + Sessions are to be traced.

Conditions**INVREQ**

RESP2 values:

- + 1 CONNSTATUS (or ACQSTATUS) was specified for a non-APPC connection.
- + 2 The CONNSTATUS (or ACQSTATUS) value or the SERVSTATUS value would result in a connection that was both ACQUIRED and OUTSERVICE.
- + 3 CONNSTATUS (or ACQSTATUS) has an invalid CVDA value.
- + 4 SERVSTATUS has an invalid CVDA value.
- + 5 NOTPENDING was specified for a non-APPC connection.
- + 6 PURGETYPE was specified for a non-VTAM connection.
- + 7 PURGETYPE has an invalid CVDA value.
- + 8 PENDSTATUS has an invalid CVDA value.
- + 11 The connection is remote.
- + 12 EXITTRACING has an invalid CVDA value.
- + 13 ZCPTRACING has an invalid CVDA value.
- + 14 EXITTRACING or ZCPTRACING was specified for a non-VTAM connection or VTAM is not installed.
- + 16 The connection is defined as an indirect link.
- + 17 CONNSTATUS (or ACQSTATUS) was specified, but CICS was initialized without support ISC support (ISC=NO).
- + 18 NOTPENDING was specified for a connection which has completed XLN processing.
- + 19 ACQUIRED was specified for a connection in FREEING status.
- + 22 Other options were specified with CANCEL or FORCECANCEL.
- + 23 The CONNECTION value is the name of the local CICS system, not a CONNECTION definition.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

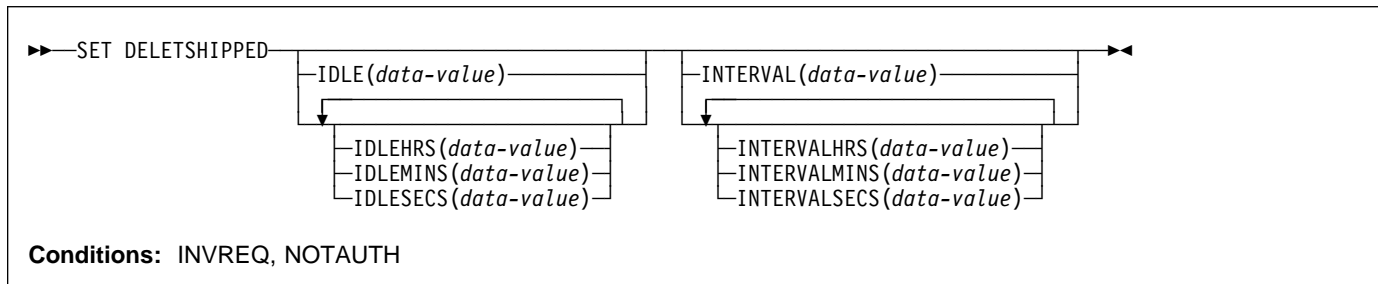
SYSIDERR

RESP2 values:

- + 9 The connection could not be found.

SET DELETSHIPED

+ Change the system settings that control automatic deletion of shipped terminal definitions.



Description

+ The SET DELETSHIPED command allows you to change
 + values that control the timeout mechanism that CICS
 + provides for deleting definitions of shipped terminals that are
 + inactive. A shipped definition is inactive if the terminal has
 + not been used locally for a specified period of time and no
 + task is waiting to be attached which requires the terminal.
 + For more information about shipped definitions, see the
 + *CICS/ESA Intercommunication Guide* and *CICS/ESA*
 + *Resource Definition Guide* manuals.

+ You can change both the length of time a shipped terminal
 + must remain inactive before being eligible for deletion (IDLE
 + time), and the interval at which CICS checks for such
 + terminals (the INTERVAL). Time values can be expressed in
 + several different ways:

- + • A 4-byte packed decimal composite, in the format
 + **0hhmmss+**, where the hours (hh) are in the range 0–99,
 + and minutes (mm) and seconds (ss) are both from 0–59.
 + Use the IDLE and INTERVAL options for this format.
- + • With separate values for hours, minutes and seconds.
 + Use IDLEHRS, IDLEMINS, and IDLESECS instead of
 + IDLE for this format, and INTERVALHRS,
 + INTERVLMINS, and INTERVALSECS instead of
 + INTERVAL. You can use any combination of hours,
 + minutes, and seconds. If you use only one, the time
 + value must be *less* than 100 hours, so that the range for
 + hours is 0–99, the range for minutes is 0–5999, and the
 + range for seconds is 0–359999. If you use two or three,
 + the range is the same for hours, but minutes and
 + seconds must both be in the range 0–59.

+ For example, to specify an IDLE time of 1 hour and 15
 + minutes, you could use any of the following:

- + IDLE(011500)
- + IDLEHRS(1) IDLEMINS(15)
- + IDLEMINS(75)
- + IDLESECS(4500).

Options

IDLE(data-value)

specifies the idle time, as a 4-byte packed decimal value
 in the form “0hhmmss+”. Idle time is the minimum time
 that a terminal must be inactive to be eligible for
 deletion.

- + See the notes at the beginning of this command
- + description for the range of values allowed.

IDLEHRS(data-value)

- + specifies, as a fullword binary value, the idle time in
- + hours (when used alone) or the hours component of the
- + idle time (when used with IDLEMINS or IDLESECS).
- + See the IDLE option.

IDLEMINS(data-value)

- + specifies, as a fullword binary value, the idle time in
- + minutes (when used alone) or the minutes component of
- + the idle time (when used with IDLEHRS or IDLESECS).
- + See the IDLE option.

IDLESECS(data-value)

- + specifies, as a fullword binary value, the idle time in
- + seconds (when used alone) or the seconds component
- + of the idle time (when used with IDLEHRS or
- + IDLEMINS). See the IDLE option.

INTERVAL(data-value)

- + specifies, as a 4-byte packed decimal value in the form
- + “0hhmmss+”, the interval between invocations of the
- + timeout delete mechanism.

+ When you change the checking interval, the next interval
 + is measured from *the time the command is issued*, **not**
 + from the previous invocation or CICS startup. If you
 + want immediate deletion, use the PERFORM
 + DELETSHIPED command, described on page 139.

- + See the notes at the beginning of this command
- + description for the range of values allowed.

INTERVALHRS(data-value)

- + specifies, as a fullword binary value, the invocation
- + interval in hours (when used alone) or the hours
- + component of the interval (when used with IDLEMINS or
- + IDLESECS). See the INTERVAL option.

INTERVALMINS(*data-value*)

- + specifies, as a fullword binary value, the invocation
- + interval in minutes (when used alone) or the minutes
- + component of the interval (when used with
- + INTERVALHRS or INTERVALSECS). See the
- + INTERVAL option.

INTERVALSECS(*data-value*)

- + specifies, as a fullword binary value, the invocation
- + interval in seconds (when used alone) or the seconds
- + component of the interval (when used with
- + INTERVALHRS or INTERVALMINS). See the
- + INTERVAL option.

Conditions**INVREQ**

RESP2 values:

- 1 The INTERVAL value is invalid.
- 2 The INTERVALHRS value is not in the range 0–99.
- 3 The INTERVALMINS value is invalid.
- 4 The INTERVALSECS value is invalid.
- 5 The IDLE value is invalid.
- 6 The IDLEHRS value is not in the range 0–99.
- 7 The IDLEMINS value is invalid.
- 8 The IDLESECS value is invalid.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

+ longer needed. Data set name blocks are not
 + removed when a data set is closed, or when CICS
 + is shut down (they are removed automatically only
 + during a cold start). If not removed, unwanted
 + data set name blocks can use up excessive
 + amounts of dynamic storage, leading to a
 + short-on-storage condition. See “Examples” for an
 + illustration of how you can identify and remove
 + unwanted data set name blocks.

BKOUTSTATUS(*cvda*) (VSAM only)

+ specifies the backout status value which CICS is to
 + assign to the data set. CVDA values are:

FAILEDDBKOUT

+ Backout is required.

NORMALBKOUT

+ Backout has been completed or was not
 + necessary.

+ **Note:** You cannot apply this option to a data set that is
 + currently in FAILINGBKOUT status.

DSNAME(*data-value*)

+ specifies the 44-character name of the data set whose
 + status is to be changed.

Conditions

DSNNOTFOUND

RESP2 values:

- + **1** CICS does not have a control block for this data set.
- + **15** CICS does not have a control block for this data set (this value occurs only when RECOVERED is specified).

INVREQ

RESP2 values:

- 2** BKOUTSTATUS has an invalid CVDA value.
- 3** ACTION has an invalid CVDA value.
- + **4** BKOUTSTATUS was specified for a data set not opened during this CICS session.
- + **5** NORMALBKOUT has been requested for a data set that is in a FAILINGBKOUT state.
- 6** FAILEDDBKOUT has been requested for a data set that is in a FAILINGBKOUT state.
- + **7** BKOUTSTATUS was specified for a BDAM data set.
- + **8** BKOUTSTATUS was specified for a path.
- + **9** BKOUTSTATUS was specified for a data set to which an open file refers.
- + **10** REMOVE was specified for a data set to which a FILE definition refers.
- + **11** REMOVE was specified for a VSAM data set not in NORMALBKOUT status.
- + **12** REMOVE was specified with BKOUTSTATUS.
- + **13** REMOVE was specified for a data set in use by another INQUIRE or SET DSNAME command, or by CICS file control processing.

- + **14** One of the MVS components required for BWO is down-level.
- + **16** RECOVERED was specified for a data set not opened during this CICS session.
- + **17** RECOVERED was specified for a BDAM data set or a VSAM path.
- + **18** RECOVERED was specified for a data set to which an open file refers.
- + **19** RECOVERED was specified for a data set not eligible for BWO processing.

IOERR

RESP2 values:

- + **20** RECOVERED was specified, but an error occurred accessing the VSAM catalog. Ensure that the data set is on an SMS-managed DASD and is known to the SMS subsystem.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

+ Examples

+ It is possible in CICS to create VSAM data sets online for
 + temporary use, and which are dynamically allocated by CICS
 + file control. Typically, this involves re-using the same file
 + control entry and setting the new temporary data set name
 + each time you need to use a new data set. This practice can
 + lead to a large number of data set name blocks occupying
 + CICS dynamic storage. These can only be removed by a
 + SET DSNAME(...) REMOVE command, or by a cold start of
 + CICS.

+ Ideally, an application that creates and uses a temporary
 + data set should explicitly delete the DSN block when it no
 + longer needs the data set. This involves two actions:

- + 1. Breaking the association between the CICS file and the data set by issuing an EXEC CICS SET FILE(...) CLOSED DISABLED command, followed by an EXEC CICS SET FILE command to set the DSNAME operand to a null value.
- + 2. Removing the data set name block by issuing an EXEC CICS SET DSNAME(...) REMOVE command.

+ To set the DSNAME to null you must code the CICS
 + commands as shown in the following examples to ensure
 + they translate and compile correctly.

SET DUMPDS

+ Assembler example

```
+*          Remove DSN block from CICS storage
+*
+*ASM XOPTS(SP)
+DFHEISTG DSECT
+TEMPDSN DS    CL44
+REMOVE    CSECT
+          PRINT GEN
+*          Find name of temporary data set if not known
+EXEC CICS INQUIRE FILE('TEMPFILE') DSNAME(TEMPDSN)
+*          Close file temporary file and set DSN to null
+EXEC CICS SET FILE('TEMPFILE') CLOSED DISABLED
+EXEC CICS SET FILE('TEMPFILE') DSNAME(=X'00')
+*          Remove DSN block from storage
+EXEC CICS SET DSNAME(TEMPDSN) REMOVE
+*
+*          Return and end
+*
+RETURN    DS    0H
+          EXEC CICS RETURN
+          END
```

+ PL/I example

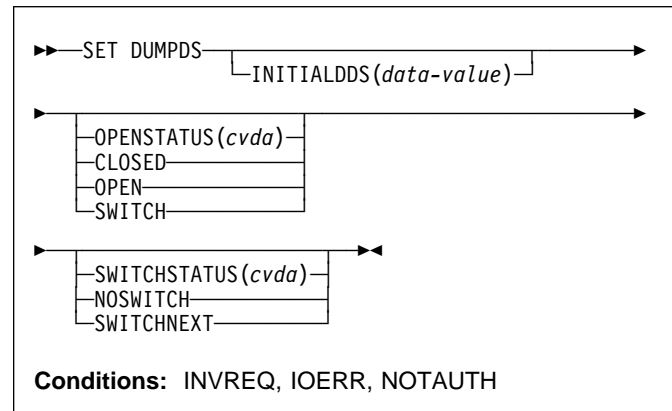
```
+*PROCESS XOPTS(SP);
+ REMOVE:PROC OPTIONS(MAIN);
+ DCL PLIXOPT STATIC EXTERNAL CHAR(10) VAR INIT('NOSTAE');
+ DCL TEMPDSN CHAR(44);
+/*
+/*          Find name of temporary data set if not known */
+EXEC CICS INQUIRE FILE('TEMPFILE') DSNAME(TEMPDSN);
+/*          Close file temporary file and set DSN to null */
+EXEC CICS SET FILE('TEMPFILE') CLOSED DISABLED;
+EXEC CICS SET FILE('TEMPFILE') DSNAME('00'X);
+/*          Remove DSN block from storage */
+EXEC CICS SET DSNAME(TEMPDSN) REMOVE
+/*
+/*          Return and end */
+/*
+EXEC CICS RETURN;
+END;
```

+ COBOL example

```
+ CBL XOPTS(SP)
+ IDENTIFICATION DIVISION.
+ PROGRAM-ID. REMOVE.
+ ENVIRONMENT DIVISION.
+ DATA DIVISION.
+ WORKING-STORAGE SECTION.
+ 77 TEMPDSN PIC X(44).
+ PROCEDURE DIVISION.
+ * Find name of temporary data set if not known
+ EXEC CICS INQUIRE FILE('TEMPFILE') DSNAME(TEMPDSN)
+ END-EXEC.
+ * Close file temporary file and set DSN to null
+ EXEC CICS SET FILE('TEMPFILE') CLOSED DISABLED
+ END-EXEC.
+ EXEC CICS SET FILE('TEMPFILE') DSNAME(LOW-VALUES)
+ END-EXEC.
+ * Remove DSN block from storage
+ EXEC CICS SET DSNAME(TEMPDSN) REMOVE END-EXEC.
+ *
+ * Return and end
+ EXEC CICS RETURN END-EXEC.
+ GOBACK.
```

SET DUMPDS

+ Change the status of the transaction dump data sets.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

- + The SET DUMPDS command allows you to change the status of CICS transaction dump data sets. Normally, either there is one of these, known as the ‘A’ dump data set, or there are two, ‘A’ and ‘B’. One is “active” (receiving dumps) and the other, if there are two, is “inactive” (standby).
- + Specifically, you can:
 - + • Open or close the active data set
 - + • Switch the roles of the active and standby data sets
 - + • Request CICS to switch automatically when the active data set is full
 - + • Specify which data set will be active the next time CICS is initialized.

+ **Note:** It is possible to bring up a CICS system without any transaction dump data sets. In this situation, only the last two functions are available.

+ Control does not return to the task issuing the command until the requested change has been made.

Options

INITIALDDS(data-value)

- + specifies, as a 1-character value, which dump data set is to be active first on subsequent warm or emergency restarts. This value is recorded in the CICS global catalog and overrides the previous value, which is set initially by the DUMPDS system initialization option.
- + The values permitted are A, B, and X. X means that CICS is to use the data set that was not active when CICS last terminated (normally or abnormally); it corresponds to the AUTO setting for the DUMPDS

- + option. (See the *CICS/ESA System Definition Guide* for a description of this option.)

OPENSTATUS(*cvda*)

- + specifies actions to be taken on the transaction dump data sets. CVDA values are:

CLOSED

- + The active CICS dump data set is to be closed.

OPEN

The active CICS dump data set is to be opened.

SWITCH

- + The roles of the dump data sets are to be
- + switched, if there are two. The data set that is
- + currently active is to become standby, and closed
- + if it is open. The current standby is to become the
- + active data set, and opened if closed.
- + If you attempt to change the open status of a data
- + set that does not exist, an IOERR exception
- + condition occurs. This can happen if you specify
- + SWITCH when there is only one dump data set, or
- + if you specify any OPENSTATUS value when
- + there are no dump data sets.

SWITCHSTATUS(*cvda*)

- + specifies whether CICS is to switch active data sets
- + automatically the next time the current dump data set
- + fills. The SWITCHSTATUS value is recorded in the
- + CICS global catalog, and therefore is remembered over
- + warm and emergency restarts. (It is set initially by the
- + DUMPSW system initialization option, described in the
- + *CICS/ESA System Definition Guide*.) An automatic
- + switch occurs only once; another SET DUMPDS
- + SWITCHNEXT command is required after each switch to
- + maintain automatic switching. CVDA values are:

NOSWITCH

- + The data sets are not be switched.

SWITCHNEXT

- + The data sets are to be switched. (SWITCHNEXT
- + has no effect unless there are two dump data sets
- + at the time the active one fills.)

Conditions**INVREQ**

RESP2 values:

- 1 INITIALDDS has an invalid value.
- 2 SWITCHSTATUS has an invalid CVDA value.
- 3 OPENSTATUS has an invalid CVDA value.

IOERR

RESP2 values:

- + 4 OPEN or SWITCH caused an error opening a data
- + set.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

+ Examples

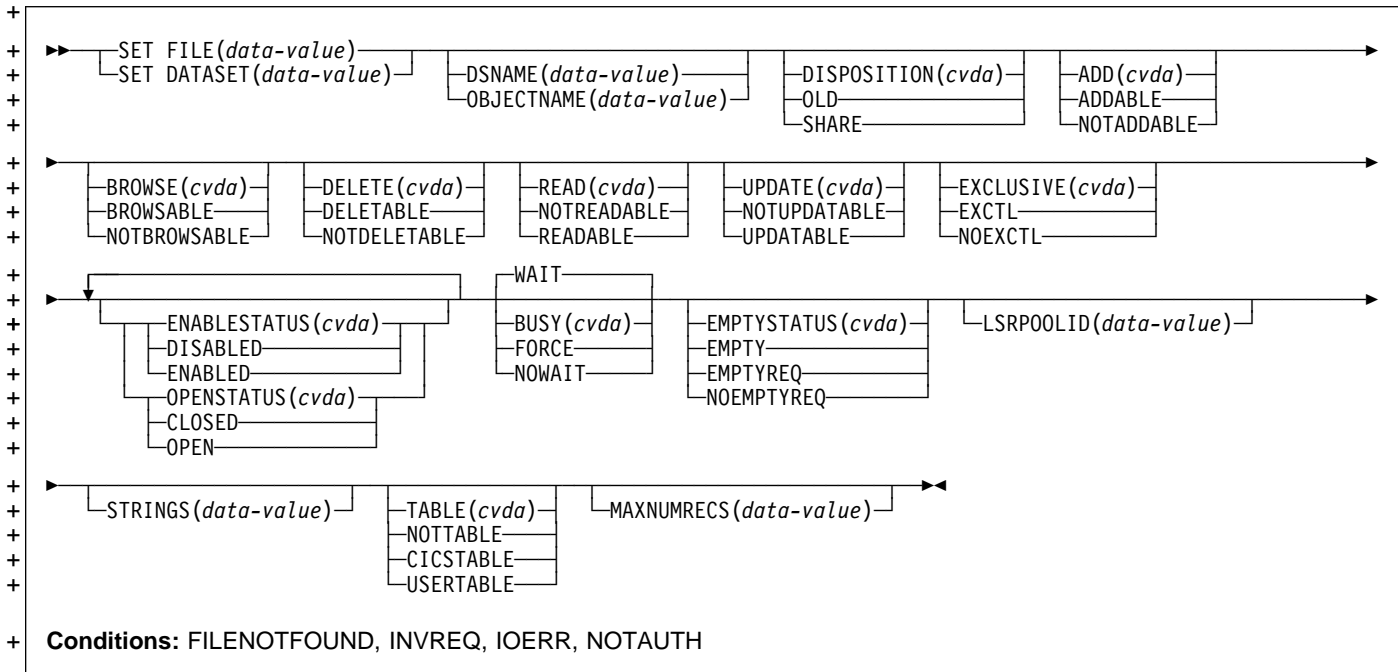
```
EXEC CICS SET DUMPDS
        INITIALDDS('A')
        SWITCH
        NOSWITCH
```

- + This example tells CICS that the A dump data set is to be
- + active first on subsequent warm and emergency restarts.
- + The OPENSTATUS setting of SWITCH makes the currently
- + active dump data set inactive, and the currently inactive
- + dump data set active. The NOSWITCH option tells CICS
- + that when the (new) active dump data set is full, there is to
- + be no automatic switch to the inactive dump data set.

SET FILE

SET FILE

+ Change the definition or status of a file.



Note: This command replaces the SET DATASET command. For compatibility, the translator accepts the keywords DATASET for FILE, OBJECTNAME for DSNAME and EMPTY for EMPTYREQ, but you should use the new keywords in new applications. For more information about the use of CVDAs, see "CICS-value data areas (CVDA)" on page 7.

Description

The SET FILE command allows you to change some elements of both the definition and the current status of a file. You can use it for any FILE definition installed on your CICS system except one defined as remote.

+ CICS requires the file to have an OPENSTATUS value of CLOSED and an ENABLESTATUS of either DISABLED or UNENABLED before changing the value of any other option. You can combine changes to OPENSTATUS and ENABLESTATUS with those to other options in most cases, however, because CICS applies option values in the following order if they are present: CLOSED, DISABLED, others, OPEN, ENABLED.

Options

ADD(*cvda*)

+ specifies whether records can be added to the file. CVDA values are:

ADDABLE

+ New records are to be allowed. (ADDABLE implies READABLE; see the READ option.)

NOTADDABLE

+ New records are not to be allowed.

BROWSE(*cvda*)

+ specifies whether the file can be browsed (read sequentially). CVDA values are:

BROWSABLE

+ Browsing is to be allowed. (BROWSABLE implies READABLE; see the READ option.)

NOTBROWSABLE

+ Browsing is not to be allowed.

BUSY(*cvda*)

+ specifies what CICS is to do if tasks that have used the file are still in execution when you issue a SET FILE command with CLOSED or DISABLED specified. (The BUSY option is ignored unless one of these options is present.) CVDA values are:

FORCE

+ CICS is to terminate abnormally all tasks using the file, close or disable it as requested, and then return control to the task that issued the command.

+ **Note:** The FORCE option terminates tasks with the CICS mechanism used for a SET TASK FORCEPURGE command. Data integrity is not guaranteed, and under certain circumstances,

+ CICS itself can terminate abnormally. See page
+ 179 for details and precautions about using this
+ option.

NOWAIT

+ NOWAIT has the same effect as WAIT, except
+ that CICS returns control to the task that issued
+ the command immediately and processes the SET
+ FILE request asynchronously.

WAIT

+ CICS is to quiesce activity on the file, close or
+ disable it as requested, and then return control to
+ the task that issued the command. During
+ quiesce, tasks that have used the file already are
+ allowed to complete, but tasks that have not yet
+ used it are denied access.

Note: CICS rejects both FORCE and WAIT if the task
issuing the SET FILE command has used the file,
because FORCE would abend the task, and WAIT would
cause a permanent wait.

DELETE(*cvda*) (VSAM only)

specifies whether records can be deleted from the file.
CVDA values are:

DELETABLE

+ Deletions are to be allowed. (DELETABLE implies
+ READABLE; see the READ option.)

NOTDELETABLE

+ Deletions are not to be allowed.

DISPOSITION(*cvda*)

+ specifies the operating system disposition value for the
+ data set to which this file refers. If there is a JCL DD
+ statement for this file, the disposition value there takes
+ precedence over the value in the FILE definition, so that
+ changing it has no effect during the current execution of
+ CICS. CVDA values are:

OLD The disposition value is to be OLD.

SHARE

The disposition value is to be SHARE.

DSNAME(*data-value*)

+ specifies the 44-character name of the VSAM object or
+ BDAM data set associated with this FILE definition.
+ CICS allocates this data set dynamically when the file is
+ opened if there is no JCL DD statement for the file. If
+ there is a DD statement, the data set name there takes
+ precedence, and this option has no effect during the
+ current execution of CICS, except to enable a SET
+ DSNAME REMOVE command (see page 156).

+ **Note:** If the file is no longer needed, but you do not
+ wish to discard the definition entirely, you can specify a
+ DSNAME that starts with a null character (X'00'). This
+ value signals CICS not to build control blocks that it
+ ordinarily builds for the data set associated with a file.

EMPTYSTATUS(*cvda*) (VSAM only)

+ specifies whether the data set to which the file refers
+ should be initialized to empty when the file is opened. It
+ is meaningful only if the associated data set is defined
+ as reusable. CVDA values are:

EMPTY

+ specifies the same value as EMPTYREQ and is
+ retained only for compatibility. Use EMPTYREQ
+ instead in new applications. This does not apply
+ to files operating in RLS mode.

EMPTYREQ

+ The data set is to be set to empty.

+ **Note:** If you specify this value for a file which
+ points to a nonreusable data set, CICS raises an
+ exception when the file is next opened. You
+ cannot specify the EMPTYREQ option on the
+ EXEC CICS SET FILE command for files that are
+ opened in RLS mode.

NOEMPTYREQ

+ The data is not to be set to empty.

ENABLESTATUS(*cvda*)

specifies whether application programs can access the
file. CVDA values are:

DISABLED

+ The file is to be unavailable.

+ **Note:** Setting a file DISABLED prevents any task
+ from using the file that has not used it already. If
+ there are tasks that have used the file, the BUSY
+ option determines whether CICS will permit them
+ to complete and when the task that issued the
+ SET FILE command will regain control (see page
+ 160).

ENABLED

+ The file is to be available.

EXCLUSIVE(*cvda*) (BDAM only)

specifies whether records on this file should be placed
under exclusive control when a read for update is
issued. CVDA values are:

EXCTL

+ Records are to be under exclusive control.

NOEXCTL

+ Records are not to be under exclusive control.

FILE(*data-value*)

+ specifies the 8-character name of the file whose
+ definition or status is to be changed.

LSRPOOLID(*data-value*) (VSAM only)

+ specifies, as a fullword binary value, whether the file is
+ to use VSAM local shared resources (LSR) and if so, the
+ number of the pool to which it is to belong. Pool
+ numbers are in the range 1–8. A value of zero means
+ that the file is not to share buffers. (Data tables must
+ use LSR, and thus zero is not allowed for a table.)

SET FILE

MAXNUMRECS(*data-value*) (VSAM only)

- + specifies, as a fullword binary value, the maximum number of records the data table for this file can hold.
- + The value must be in the range 16 to 16777215 (16MB-1).
- + **Note:** MAXNUMRECS is meaningful only for data tables (files defined with a TABLE value of CICSTABLE or USERTABLE), and consequently, an INQUIRE FILE command on a file that is not a table may return a MAXNUMRECS value of zero, even though you are not allowed to set the value to zero.

OPENSTATUS(*cvda*)

- + specifies whether the data set associated with the file is to be open or closed. CVDA values are:

CLOSED

- + The data set is to be closed.
- + **Note:** Setting a file CLOSED prevents any task from using the file that has not used it already. If there are tasks that have used the file, the BUSY option determines whether CICS will permit them to complete and also when the task that issued the SET FILE command will regain control (see page 160).

OPEN

- + The data set is to be open.

READ(*cvda*)

- + specifies whether records can be read from the file when no other processing options are allowed (reading is automatically permitted if the file is defined as ADDABLE, BROWSABLE, DELETABLE, or UPDATABLE, even when NOTREADABLE). CVDA values are:

NOTREADABLE

- + Read requests are to be rejected.

READABLE

- + Read requests are to be allowed.

STRINGS(*data-value*) (VSAM only)

specifies, as a fullword binary value, the maximum number of concurrent operations to allow on this file, in the range 1-255.

TABLE(*cvda*) (VSAM only)

- + specifies whether the file is to be a data table and if so, the type of table. CVDA values are:

CICSTABLE

- + The file is to be a CICS-maintained data table.

NOTTABLE

- + The file is not to be a data table.

USERTABLE

- + The file is to be a user-maintained data table. The record format for the file must be VARIABLE if you specify this value.

- + **Note:** MAXNUMRECS must be greater than zero for either a CICS- or user-maintained table.

UPDATE(*cvda*)

- + specifies whether records in the file can be updated (rewritten). CVDA values are:
- + NOTUPDATABLE
Updates are not to be allowed.
- + UPDATABLE
Updates are to be allowed. (UPDATABLE implies READABLE; see the READ option.)

Conditions

FILENOTFOUND

RESP2 values:

- + **18** The file cannot be found.

INVREQ

RESP2 values:

- + **1** The file is remote.
- + **2** The file is not closed.
- + **3** The file is not in DISABLED or UNENABLED status.
- + **4** ADD has an invalid CVDA value.
- + **5** BROWSE has an invalid CVDA value.
- + **6** BUSY has an invalid CVDA value.
- + **7** DELETE has an invalid CVDA value.
- + **8** DISPOSITION has an invalid CVDA value.
- + **9** EMPTYSTATUS has an invalid CVDA value.
- + **10** LSRPOOLID was specified for a non-VSAM file.
- + **11** The LSRPOOLID value is out of range.
- + **12** READ has an invalid CVDA value.
- + **13** The STRINGS value is out of range, or the file is non-VSAM.
- + **14** UPDATE has an invalid CVDA value.
- + **15** The data set associated with the file cannot be opened because it has had a backout failure.
- + **16** OPENSTATUS has an invalid CVDA value.
- + **17** ENABLESTATUS has an invalid CVDA value.
- + **19** DELETE was specified for a non-VSAM file.
- + **20** EMPTYSTATUS was specified for a non-VSAM file.
- + **21** CLOSED or DISABLED has been specified without NOWAIT by a task that has issued one or more recoverable requests within the current unit of work.
- + **22** ENABLED was specified for a file that is currently in DISABLING or UNENABLING status.
- + **23** EXCLUSIVE has an invalid CVDA value.
- + **24** EXCLUSIVE was specified for a non-BDAM file.
- + **28** OPEN, CLOSE, ENABLE, or DISABLE was specified but a global user exit running at exit point XFCSREQ instructed CICS not to carry out the command.
- + **29** TABLE has an invalid CVDA value.
- + **30** The MAXNUMRECS value is out of range.
- + **31** TABLE was specified for a non-VSAM file.

- + **34** MAXNUMRECS was specified for a non-VSAM file.
- + **37** The TABLE or LSRPOOLID value will result in an inconsistent combination.
- + **38** The TABLE or MAXNUMRECS value will result in an inconsistent combination.
- + **39** USERTABLE was specified but the record format is not VARIABLE.

IOERR

RESP2 values:

- + * OPEN has failed in VSAM; the RESP2 field contains the VSAM response code.
- + * CLOSE has failed in VSAM; the RESP2 field contains the VSAM response code.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.
- 101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

Examples

```
EXEC CICS SET FILE ('FILE12')
      WAIT
      CLOSED
      DISABLED
      DELETABLE
+     LSRPOOLID(7)
+     STRINGS(50)
```

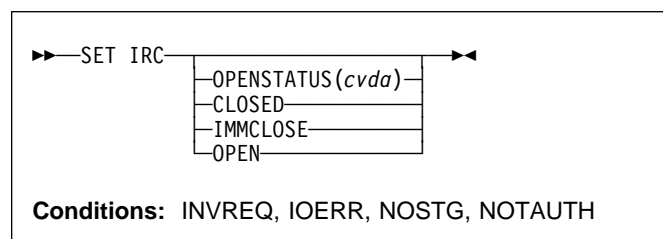
```
EXEC CICS SET FILE ('FILE12')
      OPEN
      ENABLED
```

+ The first command tells CICS to disable the file FILE12, close it, and then change its definition to permit deletions, use LSR pool 7, and allow up to 50 concurrent operations. If tasks are using the file at the time of the SET command, CICS will first set the ENABLESTATUS to DISABLING and allow these tasks to finish before completing the command and returning control to the task that issued it.

+ The second SET FILE command reopens the file and makes it available for use.

SET IRC

Open or close interregion communication.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

- + The SET IRC command allows you to start (open) or stop (close) interregion communication (IRC) in your CICS region.
- + IRC must be open for your region to communicate with another CICS region using a multiregion operation (MRO) connection or for a non-CICS client region to use your CICS over an external CICS interface (EXCI) connection.

- + Support for this type of communication must be specified at CICS startup (in the ISC initialization option), and at least one CONNECTION resource must be defined with an ACCESSMETHOD value indicating MRO; otherwise exception conditions occur when you attempt to open IRC.
- + The *CICS/ESA Intercommunication Guide* describes the various requirements.

Options**OPENSTATUS(cvda)**

- + specifies whether IRC communications should be started (open) or stopped (closed), and if CICS needs to stop IRC, whether tasks using MRO should be allowed to complete first. CVDA values are:

CLOSED

- + IRC is to be stopped. If it is currently open, CICS is to quiesce all MRO activity and then close IRC. Tasks using CICS-to-CICS MRO sessions and EXCI sessions are allowed to complete before closure, but new tasks requiring IRC are not begun.

IMMCLOSE

- + IRC is to be stopped. If currently open, CICS is to terminate abnormally any tasks using IRC immediately and then close IRC.

OPEN

- + IRC is to be started. If currently closed, CICS is to open it.

SET JOURNALNUM

Conditions

INVREQ

RESP2 values:

- + 1 A program required for IRC, DFHCRSP, is unavailable.
- + 2 OPENSTATUS has an invalid CVDA value.
- + 4 CICS was initialized without IRC support (ISC=NO).
- + 5 No connection has been defined.
- + 6 The VTAM APPLID for this CICS is blanks; IRC requires a non-blank APPLID.
- + 7 Another CICS using IRC has the same VTAM APPLID as this one; unique names are required.
- + 8 IRC rejected the open of this CICS because it had already reached the maximum number of logons.
- + 18 IRC support (the DFHIRP module) is below the level required by this CICS system.

IOERR

RESP2 values:

- + 12 IRC initialization failed.
- + 13 The logon to IRC failed.
- + 14 An attempt to attach the node error transaction, CSNC, failed.
- + 15 An error occurred closing IRC.

NOSTG

RESP2 values:

- + 9 CICS storage is insufficient for the request.
- + 10 MVS storage is insufficient (SVC block request rejected).
- + 11 MVS storage is insufficient (SUBSYS block request rejected).

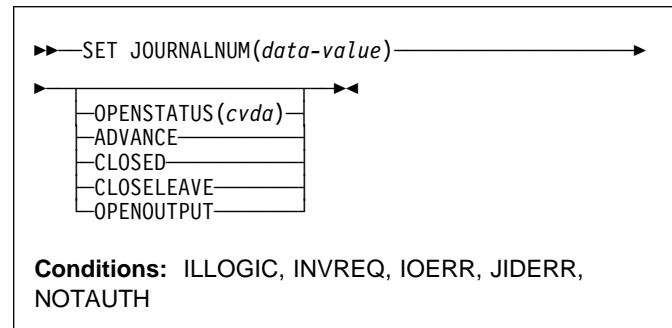
NOTAUTH

RESP2 values:

- + 100 The user associated with the issuing task is not authorized to use this command.

SET JOURNALNUM

+ Open or close a journal, or change the current data set.



For more information about the use of CVDAs, see "CICS-value data areas (CVDA)" on page 7.

Description

+ The SET JOURNALNUM command allows you to open or close a journal, to switch or reset a disk journal, or to change the volume of a tape journal. The options that apply depend on the type of the journal, as noted in the descriptions below. + See the JTYPE option of the INQUIRE JOURNALNUM or the *CICS/ESA Resource Definition Guide* for more information about journal types.

+ SET JOURNALNUM commands apply to the system log (Journal 1) as well as installation-defined journals. However, you should not set the log CLOSED, because any task that writes to the log thereafter will fail. If CICS attempts to back out the failing task, it will not be able to write the required records to the log, and will abend. If you need to change the data set being used for the log, use the ADVANCE option instead.

Options

+ **JOURNALNUM**(data-value)

+ specifies, as a halfword binary value, the number of the journal that you are changing. Journal numbers are in the range 1–99. Journal 1 is the CICS system log.

+ **OPENSTATUS**(cvda)

+ specifies the change to be made to the journal. CVDA values are:

ADVANCE

+ If the journal is on tape, CICS closes the current volume and opens the next one.

+ For a DISK2 journal, the roles of the two data sets are switched. The data set that CICS is currently writing is closed and becomes the standby, and the standby is opened and becomes the current data set.

- + For a DISK1 journal, the data set is closed and reopened, so that subsequent writes to the journal reuse the data set from the beginning.
- + ADVANCE cannot be used for an SMF journal and causes an exception condition.
- + In all cases, the journal must be open when ADVANCE is requested.

CLOSED

- + The journal is closed, if open. If it is a tape journal, the current volume is rewind.

CLOSELEAVE

- + The journal is closed, if open. CLOSELEAVE has the same effect as CLOSED for a disk journal, but for a tape journal, it does not rewind the volume.

OPENOUTPUT

- + The journal is opened for output, if not already open. If you have specified deferred opening in a definition of a journal, SET JOURNALNUM OPENOUTPUT can be used to open it, which must be done for the journal to be used in CICS.

Conditions

ILLOGIC

RESP2 values:

- + **6** ADVANCE is specified for a journal that is not open.

INVREQ

RESP2 values:

- + **2** OPENSTATUS has an invalid CVDA value.
- + **3** The OPENSTATUS value is not valid for this type of journal.

IOERR

RESP2 values:

- 5** An error has occurred in the device, the medium, or the access method.
- 7** Journal archiving control data set (JACD) error.
- 8** Journal archiving control data set (JACD) logic error.

JIDERR

RESP2 values:

- + **1** The journal cannot be found.

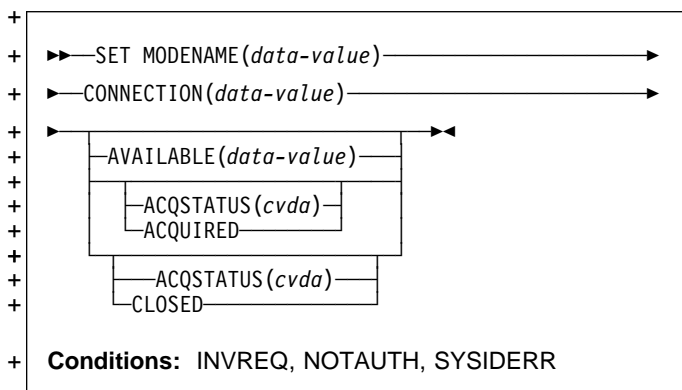
NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.
- 101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

SET MODENAME

- + Change the number of sessions in an APPC session group.



- + For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The SET MODENAME command enables you to increase or decrease the number of sessions **available** (bound) in a session group on a particular APPC connection. You identify the group to be changed by the MODENAME and CONNECTION values in its SESSIONS definition, rather than the name of the SESSIONS definition. You need both values because MODENAMES are not necessarily unique across connections.

SET MODENAME applies only to parallel session groups on an APPC connection on which CICS is already in session with its partner system, and only to groups created with a SESSIONS resource definition (not to SNASVCMG LU services manager sessions). The changes last only until the connection is released or the number of sessions is changed again.

If you increase the number of sessions, you can specify whether or not CICS should acquire the additional sessions; if you decrease the number, CICS unbinds the excess sessions automatically. If more than the target number of sessions are in use at the time of the command, CICS allows activity to quiesce before unbinding. Tasks using a session on the connection are allowed to complete, but new tasks requiring a session are not started until activity drops below the new limit.

Note: CICS uses a task that executes LU Services Manager transaction CLS1 to acquire or release sessions on parallel-session APPC connections. Data is passed to the task in a temporary storage queue whose name begins with the default prefix of DF. If your system defines queues named starting with DF as recoverable, CICS cannot initiate this task until a subsequent commit on the part of the task that issued the SET MODENAME command (either a SYNCPOINT command or an implicit syncpoint).

SET MONITOR

Options

ACQSTATUS(*cvda*)

- + specifies either that additional sessions are to be acquired if the AVAILABLE value increases the number, or that the number of available sessions is to be set to zero. CVDA values are:

ACQUIRED

- + Additional sessions, if any, are to be acquired.

CLOSED

- + The number of sessions is to be set to zero.
- + CLOSED is equivalent to specifying AVAILABLE (0) and should not be specified with AVAILABLE.
- + This value prevents either of the connected systems from using a session in the group.

AVAILABLE(*data-value*)

- + specifies, as a halfword binary value, the number of sessions which are to be available for use at any one time. The range for this value is from zero to the MAXIMUM value specified in the SESSIONS definition; you can determine this limit, if necessary, with an INQUIRE MODENAME command.

CONNECTION(*data-value*)

- + specifies the 4-character name of the connection for which this group of sessions is defined (from the CONNECTION value in the SESSIONS definition).

MODENAME(*data-value*)

- + specifies the 8-character MODENAME value of the group of sessions that you are modifying (from its SESSIONS definition).

Conditions

INVREQ

RESP2 values:

- + **3** MODENAME 'SNASVCMG' was specified.
- + **4** The AVAILABLE value is out of range.
- + **5** AVAILABLE was specified but CICS is not in session on this connection.
- + **6** CLOSED was specified with AVAILABLE.
- + **7** ACQSTATUS has an invalid CVDA value.
- + **8** This is not a parallel-session APPC group.
- + **9** ACQUIRED was specified but CICS is not in session on this connection.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

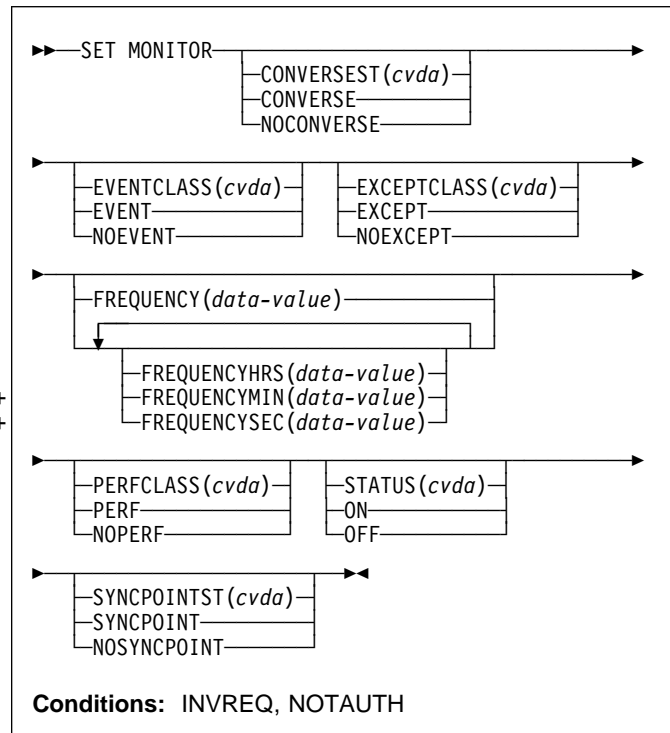
SYSIDERR

RESP2 values:

- + **1** The connection cannot be found.
- + **2** The MODENAME within the connection cannot be found.

SET MONITOR

Change CICS monitoring options.



For more information about the use of CVDA's, see "CICS-value data areas (CVDA's)" on page 7.

Description

The SET MONITOR command allows you to switch CICS monitoring on or off, to modify the settings of the monitoring options, and to select the classes of monitoring data to be recorded.

CICS monitoring is controlled by a master switch (the STATUS option). Monitor data is accumulated only while the STATUS option has the value ON, and only for tasks that **begin** while STATUS is ON.

When monitoring is active, CICS accumulates three types of data for each individual task:

- SYSEVENT data (elapsed time, obtained through MVS timing facilities)
- Performance data (types and counts of CICS commands, timings, etc.)
- Exception data (waiting for a VSAM string, for example)

Three additional switches determine which of these classes of monitor data are recorded (written out to the SMF data set). SYSEVENT data is recorded only if the EVENTCLASS option setting is EVENT, exception data only if EXCEPTCLASS is EXCEPT, and performance data only if PERFCLASS is PERF. For an individual task, class data is

recorded only if the class switch is on both at the time the task **starts** and at the time that class of data is written out.

Exception class data is written at the end of the event to which the exception corresponds. SYSEVENT class data is written at end of task. Performance class data is written only at these specific times:

- At end of task
- At a terminal-receive wait, if the CONVERSEST value is CONVERSE
- At a frequency interval, if the interval is not zero
- At a syncpoint, if the SYNCPOINTST value is SYNCPOINT
- When a user event monitoring point with the DELIVER option occurs.

If you change STATUS from ON to OFF, CICS stops accumulating and recording monitor data. Data for tasks in flight that is not already recorded is lost even if you turn monitoring back on before end of task.

+ Furthermore, if you are recording performance data, you should specify NOPERF in any command that sets monitoring OFF, to ensure that buffers containing recorded data for completed tasks are flushed; some of this data can be lost otherwise.

+ If you leave STATUS on but turn one of the recording options off and then back on during a task, however, data loss depends on the class, as follows:

- SYSEVENT data not already written is lost for the rest of the task.
- Exception data is not written out for exceptions that occur while EXCEPTCLASS is NOEXCEPT but, if you change back to EXCEPT, subsequent exceptions are recorded.
- If you change PERFCLASS from PERF to NOPERF during execution of a task, performance data already accumulated is recorded, but then recording stops. Accumulation continues, however. Therefore, if you change back to PERF before task end, no data is lost unless a monitor point with the DELIVER option occurs while NOPERF is in force. (DELIVER resets the counters.) The other conditions that ordinarily cause writing—syncpoint with a SYNCPOINTST value of SYNCPOINT, terminal receive wait with a CONVERSEST value of CONVERSE, or expiration of the frequency interval—do not reset the counts while recording is off, so that no counts are lost, although they may be combined.

Options

| CONVERSEST(*cvda*)

+ specifies how CICS is to record performance data for conversational tasks (tasks that wait for terminal or session input).

| CONVERSE

+ CICS is to produce a performance class record each time the task waits for terminal input as well as at task end, representing the part of the task since the previous wait (or task start). (Waits occur during execution of a CONVERSE command or a RECEIVE command that follows a SEND.)

| NOCONVERSE

+ CICS is to accumulate performance data across terminal waits and produce a single performance class record.

EVENTCLASS(*cvda*)

+ specifies whether the SYSEVENT class of monitoring data is to be recorded when monitoring is active. CVDA values are:

EVENT

+ SYSEVENT data is to be recorded.

NOEVENT

+ SYSEVENT data is not to be recorded.

EXCEPTCLASS(*cvda*)

+ specifies whether the exception class of monitoring data is to be recorded when monitoring is active. CVDA values are:

EXCEPT

+ Exception data is to be recorded.

NOEXCEPT

+ Exception data is not to be recorded.

| FREQUENCY(*data-value*)

+ specifies the interval at which CICS is to produce performance class records for long-running tasks. If a task runs longer than the frequency interval, CICS records its performance data separately for each interval or fraction.

+ The frequency interval can be expressed in several ways:

- A 4-byte packed decimal composite, in the format “0hhmss+”, using the FREQUENCY option.
- With separate hours, minutes, and seconds, using the FREQUENCYHRS, FREQUENCYMIN, and FREQUENCYSEC options. You can use these options singly or in any combination.

+ Whichever method you use, the interval value must be either zero or in the range from 15 minutes to 24 hours. Zero means CICS is to produce performance records only at task end, regardless of the length of the task.

+ In addition, if you use FREQUENCY or more than one of the separate options, the minutes and seconds portions of the value must not be greater than 59 (FREQUENCYMIN or FREQUENCYSEC used alone can exceed 59). For example, you could express an interval of 1 hour and 30 minutes in any of the following ways:

SET MONITOR

- + • FREQUENCY(13000)
- + • FREQUENCYHRS(1), FREQUENCYMIN(30)
- + • FREQUENCYMIN(90)
- + • FREQUENCYSEC(5400)

FREQUENCYHRS(*data-value*)

- + specifies the hours component of the frequency interval,
- + in fullword binary form (see the FREQUENCY option).

FREQUENCYMIN(*data-value*)

- + specifies the minutes component of the frequency interval, in fullword binary form (see the FREQUENCY option).

FREQUENCYSEC(*data-value*)

- + specifies the seconds component of the frequency interval, in fullword binary form (see the FREQUENCY option).

PERFCLASS(*cvda*)

- + specifies whether the performance class of monitoring data is to be recorded when monitoring is active. CVDA values are:

NOPERF

- + Performance data is not to be recorded.

PERF

- + Performance data is to be recorded.

STATUS(*cvda*)

- + specifies whether CICS monitoring is to be active or disabled. CVDA values are:

- + OFF Monitoring is not to occur. No data will be accumulated or written out, irrespective of the settings of the monitoring data classes.

- + ON Monitoring is to be active. Data will be accumulated for all classes of monitor data, and written out for those classes that are active.

| **SYNCPOINTST**(*cvda*)

- + specifies whether CICS is to record performance class data separately for each logical unit of work (LUW) within tasks that contain multiple LUWs. An LUW within a task ends when a syncpoint occurs, either explicitly (a SYNCPOINT command) or implicitly (a DL/I TERM call, for example, or task end); a new LUW begins immediately after, except at end of task. When rollback occurs on a syncpoint, the LUW does not end. CVDA values are:

NOSYNCPOINT

- + Performance data is to be combined over all LUWs in a task.

SYNCPOINT

- + Performance data is to be recorded separately for each LUW.

Conditions

INVREQ

RESP2 values:

- | | |
|----|--|
| 1 | STATUS has an invalid CVDA value. |
| 2 | PERFCLASS has an invalid CVDA value. |
| 3 | EXCEPTCLASS has an invalid CVDA value. |
| 4 | EVENTCLASS has an invalid CVDA value. |
| 5 | CONVERSEST has an invalid CVDA value. |
| 6 | SYNCPOINTST has an invalid CVDA value. |
| 7 | The FREQUENCY value is invalid. (The hours exceed 24, minutes or seconds exceed 59, or total value is out of range.) |
| 8 | The FREQUENCYHRS value is out of range. |
| 9 | The FREQUENCYMIN value is out of range. |
| 10 | The FREQUENCYSEC value is out of range. |

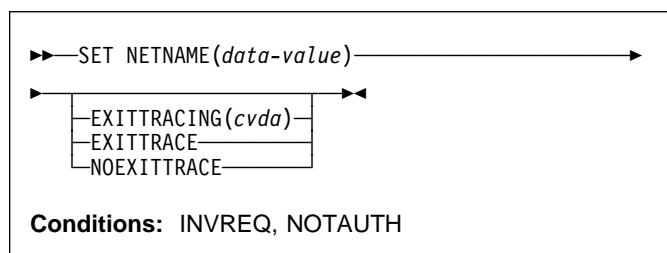
NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

SET NETNAME

- + Change the tracing of a VTAM terminal.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

- + The SET NETNAME command allows you to control CICS
- + VTAM exit tracing for a particular VTAM terminal (or
- + session). You can specify any VTAM terminal or session,
- + including one not yet installed in CICS, so that you can trace
- + the autoinstall process as well as other operations.

Options

EXITTRACING(*cvda*)

- + specifies whether this terminal (or session) should be
- + traced when CICS is tracing terminal-specific invocations
- + of its VTAM exits. (You can turn exit tracing on and off
- + with a SET TRACEFLAG TCEXITSTATUS command or
- + the CICS-supplied transaction CETR.) CVDA values
- + are:

EXITTRACE

- + The terminal is to be traced.

NOEXITTRACE

- + The terminal is not to be traced.

NETNAME(*data-value*)

- + specifies the 8-character VTAM network identifier of the
- + terminal or session for which you are specifying tracing.

Conditions

INVREQ

RESP2 values:

- + **27** EXITTRACING has an invalid CVDA value.
- + **29** The terminal is not a VTAM terminal.

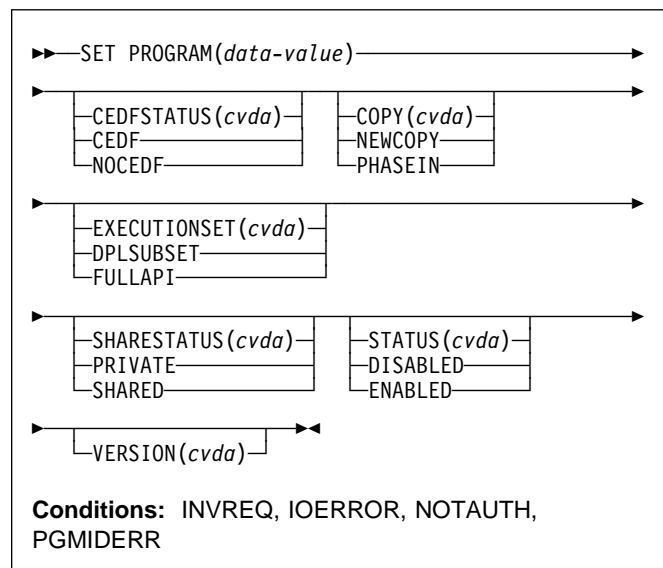
NOTAUTH

RESP2 values:

- + **100** The user associated with the issuing task is not
- + authorized to use this command.

SET PROGRAM

- + Change a PROGRAM, MAPSET, or PARTITIONSET
- + definition.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

- + The SET PROGRAM command modifies the definition of a
- + particular program, map set, or partition set installed in your
- + CICS system. All of these resources are load modules and,
- + therefore, CICS uses the same SET command for all three.
- + To avoid confusion, we use the word **module** to refer to the
- + object of your command, except when the option applies only
- + to executable programs.

Options

+ CEDFSTATUS(*cvda*) (programs only)

- + specifies what action the execution diagnostic facility
- + (EDF) is to take if this program is executed under EDF.
- + CVDA values are:

CEDF

- + EDF diagnostic screens are to be displayed. If the
- + program was translated with the EDF option, all
- + EDF screens are displayed; if it was translated
- + with NOEDF, only the program initiation and
- + termination screens appear.

NOCEDF

- + No EDF screens are displayed.

You cannot specify CEDFSTATUS for a remote program.

SET PROGRAM

COPY(*cvda*)

+ specifies that a new copy of the program is to be used the next time the module is requested. LINK, XCTL, LOAD, ENABLE and BMS commands can cause a module request.

+ CICS does not load the module at this time, but it does ensure that a copy is available. If you have specified the SHARED option and the module is in the link-pack area, the LPA copy satisfies this requirement. Otherwise, CICS searches the concatenated DFHRPL libraries, and returns an IOERR exception if it cannot locate a copy there. CVDA values are:

NEWCOPY

+ The module is to be refreshed only if it is not currently in use; otherwise CICS returns an INVREQ exception instead. (You can determine whether a module is in use from the RESCOUNT option in an INQUIRE PROGRAM command; a value of zero means the program is not in use.)

PHASEIN

+ The refresh is to occur whether or not the module is in use. If it is, the copy or copies in use remain until they are no longer in use, but all requests that occur after the refresh use the new copy.

+ COPY cannot be specified for any module currently loaded with the HOLD option, or for any program defined as remote.

+ **EXECUTIONSET**(*cvda*) (programs only)

+ specifies whether the program is to be restricted to executing the distributed program link (DPL) subset of the CICS API. EXECUTIONSET applies only to executable programs, and governs the API only when a program executes locally. (Programs are always restricted to this subset when invoked remotely—that is, when executing at or below the level of a program invoked by DPL.) CVDA values are:

DPLSUBSET

+ The program is always to be restricted. You cannot specify this value for CICS-supplied programs (those beginning with 'DFH').

FULLAPI

+ The program is not to be restricted unless invoked remotely.

PROGRAM(*data-value*)

+ specifies the 8-character name of the program, map set, or partition set definition to be changed.

SHARESTATUS(*cvda*)

+ specifies where CICS should obtain the module the next time a new copy is required. A new copy request can result from either an explicit request (SET PROGRAM COPY or the CEMT equivalent) or from a command that requires the module that is issued when CICS does not currently have a copy. CVDA values are:

PRIVATE

+ The module is to be loaded from the concatenated libraries named on the DFHRPL DD statement.

SHARED

+ The link-pack area copy is to be used, if one is available. If not, the module will be loaded as if SHARESTATUS were PRIVATE.

You cannot specify SHARESTATUS for a remote program.

STATUS(*cvda*)

+ specifies whether the module is to be available for use. CVDA values are:

ENABLED

+ The module is to be available.

DISABLED

+ The module is to be unavailable. Programs beginning with 'DFH' cannot be disabled.

+ For a program defined as remote, this option governs availability only when the program is invoked through the local CICS system; it does not change availability on the remote system.

VERSION(*cvda*)

+ returns a CVDA value indicating whether the copy CICS located for a COPY request is different from the current copy. A value is returned only when the COPY option is also specified; in other cases the CVDA value is unchanged. For this purpose, CICS defines "different" to mean a switch from a copy loaded from the DFHRPL libraries to the link-pack area copy or vice-versa, or a copy loaded from a disk location different from that of the current copy. CVDA values are:

NEWCOPY

+ The new copy is different.

OLDCOPY

+ The new copy is not different.

Conditions

INVREQ

RESP2 values:

- 1 DISABLED or DPLSUBSET was specified for a program beginning 'DFH'.
- 2 STATUS has an invalid CVDA value.
- 3 NEWCOPY was specified and RESCOUNT is not equal to zero.
- 4 SHARESTATUS has an invalid CVDA value.
- 5 COPY has an invalid CVDA value.
- + 6 COPY was specified for a module currently loaded with the HOLD option.
- 9 CEDFSTATUS has an invalid CVDA value.
- + 17 You have specified an option that is invalid for a remote program (CEDFSTATUS, COPY, EXECUTIONSET or SHARESTATUS).

- + **18** You have specified an option that is invalid for a map set (CEDFSTATUS or EXECUTIONSET).
- + **19** You have specified an option that is invalid for a partition set (CEDFSTATUS or EXECUTIONSET).
- + **20** EXECUTIONSET has an invalid CVDA value.

IOERR

RESP2 values:

- + **8** The COPY option was specified but CICS could not locate the module.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.
- 101** The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

PGMIDERR

RESP2 values:

- + **7** The program, map set, or partition set cannot be found.

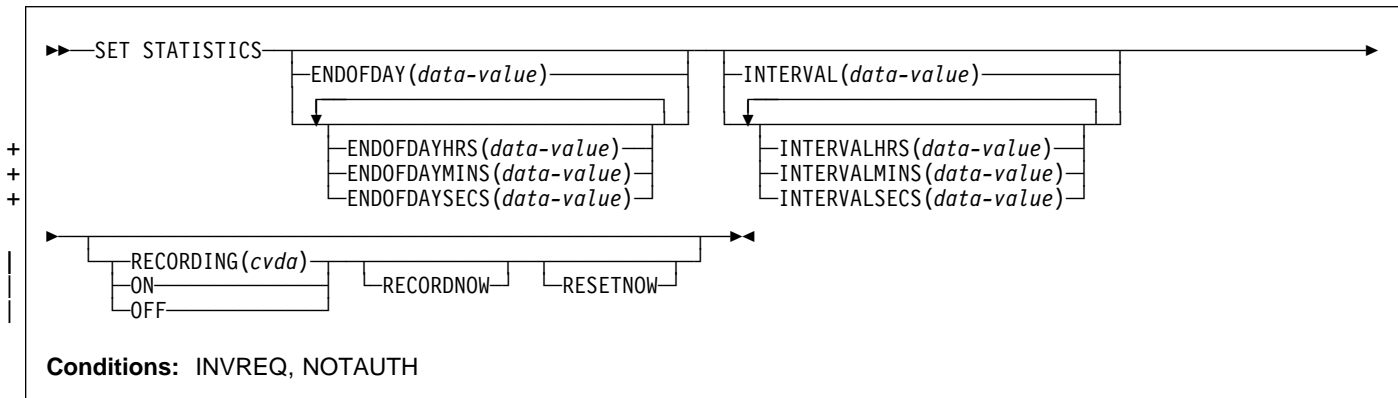
Examples

```
EXEC CICS SET PROGRAM ('PROGA')
        PHASEIN
        PRIVATE
        DISABLED
```

- + In this example, CICS is to make module PROGA
- + unavailable for new requests, and to locate a new copy in
- + one of the DFHRPL libraries. Any copies of PROGA with a
- + non-zero RESCOUNT will remain until no longer in use, but
- + new requests for PROGA will fail until PROGA is set to
- + ENABLED status. On the first request after the module is
- + enabled, CICS will load the new copy and make it the current
- + one.

SET STATISTICS

+ Change the recording of CICS statistics.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

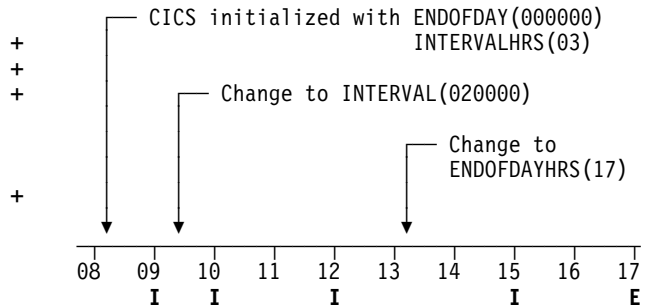
+ The SET STATISTICS command allows you to change values that control the recording of CICS statistics and to reset the counts.

+ CICS records system and resource statistics periodically if the RECORDING switch is on, at a frequency governed by the INTERVAL option. These statistics are called **interval statistics**. At end-of-day time (the ENDOFDAY option), CICS records **end-of-day statistics**—which are the statistics for the interval since the last resetting—whether or not the switch is on, ensuring that statistics are written at least once a day. Recording occurs on a system management facility (SMF) data set, and the counts are reset after recording.

+ When CICS is initialized, the length of the first interval is adjusted so that an integral number of intervals remains until end-of-day time. If you change the recording interval, the same adjustment is made to the current interval. Changing the end-of-day value, however, does not affect recording times until the new end-of-day is reached. The arrival of end-of-day time, whether changed or not, ends the current recording interval. After the statistics are written out, the next interval is adjusted again if necessary, so that the recording interval divides the time remaining to the next end-of-day evenly.

+ **Note:** These adjustments are made whether or not the statistics for the interval get recorded. Consequently, if you want to capture all of the statistics, set RECORDING ON or let your end-of-day recording cover all of them by setting the recording interval to 24 hours.

These rules are illustrated by the following example. **I** indicates an interval recording and **E** indicates an end-of-day + recording. The system is cold started with STATRCD, the + option that sets the initial value for the RECORDING switch, + set to ON.



The *CICS/ESA Performance Guide* contains more detail about CICS statistics, including the values to which various types of statistics are reinitialized.

+ The two time values that you can set with this command can be expressed in several ways:

- + • A 4-byte packed decimal composite, in the format “0hhmss+”, which you specify with the ENDOFDAY or INTERVAL option.
- + • Separate hours, minutes and seconds, which you specify with the ENDOFDAYHRS, ENDOFDAYMINS, and ENDOFDAYSECS options (instead of ENDOFDAY) and INTERVALHRS, INTERVALMINS, and INTERVALSECS (instead of INTERVAL). You can use these options singly or in any combination.

+ For example, you could express an INTERVAL of 1 hour and + 30 minutes in any of the following ways:

- + • INTERVAL(13000)
- + • INTERVALHRS(1), INTERVALMINS(30)
- + • INTERVALMINS(90)
- + • INTERVALSECS(5400)

Options

ENDOFDAY(*data-value*)

- + specifies the end-of-day time, as a 4-byte packed decimal field in the format 0hhmss+.
- + End-of-day time is expressed in local time and must be in the range 00:00:00–23:59:59. When you use the ENDOFDAY option, or more than one of the separate end-of-day options, neither the minutes nor the seconds portions can exceed 59. If you use ENDOFDAYMINS alone the limit is 1439 and for ENDOFDAYSECS used alone it is 86399.

ENDOFDAYHRS(*data-value*)

- + specifies the hours component of the end-of-day time, in fullword binary form. (See the ENDOFDAY option.)

ENDOFDAYMINS(*data-value*)

- + specifies the minutes component of the end-of-day time, in fullword binary form. (See the ENDOFDAY option.)

ENDOFDAYSECS(*data-value*)

- + specifies the seconds component of the end-of-day time, in fullword binary form. (See the ENDOFDAY option.)

INTERVAL(*data-value*)

- + specifies the recording interval for system statistics, as a 4-byte packed decimal field in the format 0hhmss+.
- + The interval must be at least a minute and no more than 24 hours. When you use the INTERVAL option, or more than one of the separate interval options, neither the minutes nor the seconds portions of the time must exceed 59. If you use INTERVALMINS alone the range is 1–1440 and for INTERVALSECS used alone it is 60–86400.

INTERVALHRS(*data-value*)

- + specifies the hours component of the recording interval, in fullword binary form. (See the INTERVAL option.)

INTERVALMINS(*data-value*)

- + specifies the minutes component of the recording interval, in fullword binary form. (See the INTERVAL option.)

INTERVALSECS(*data-value*)

- + specifies the seconds component of the recording interval, in fullword binary form. (See the INTERVAL option.)

RECORDING(*cvda*)

- # specifies whether to record the interval statistics.
- # End-of-day statistics, requested statistics and unsolicited statistics are always recorded, specifies whether interval statistics are to be recorded. irrespective of the setting of the RECORDING option." Unsolicited statistics are resource statistics, recorded when the resource is discarded. Requested statistics are those called for by a PERFORM STATISTICS RECORD command, described on page 144, or by a CEMT PERFORM STATISTICS transaction.) CVDA values are:

OFF switches off the recording of interval statistics.

ON switches on the recording of interval statistics.

RECORDNOW

- + specifies that the current statistics are to be written out immediately. The effect is the same as a PERFORM STATISTICS RECORD ALL command and, as in the case of that command, the counts are not reset unless you specify RESETNOW as well. RECORDNOW can be specified only when the RECORDING status is changed from ON to OFF or from OFF to ON.

RESETNOW

- + specifies that the statistics counters are to be reset to their initial values. The initial value for a given counter depends on the type of statistic being collected; see 'CICS statistics tables' in the *CICS/ESA Performance Guide* for specific information. The reset can be requested only when the RECORDING status is changed from ON to OFF or from OFF to ON.

Conditions

INVREQ

RESP2 values:

- + **1** The INTERVAL value is out of range.
- + **2** The ENDOFDAY value is out of range.
- + **3** RECORDING has an invalid CVDA value.
- + **4** The INTERVALHRS value is out of range.
- + **5** The INTERVALMINS value is out of range.
- + **6** The INTERVALSECS value is out of range.
- + **7** More than one of the interval values has been used and the combination either exceeds 24 hours or is less than 1 minute.
- + **8** The ENDOFDAYHRS value is out of range.
- + **9** The ENDOFDAYMINS value is out of range.
- + **10** The ENDOFDAYSECS value is out of range.
- + **11** RESETNOW or RECORDNOW has been specified, but the RECORDING value has not been changed.

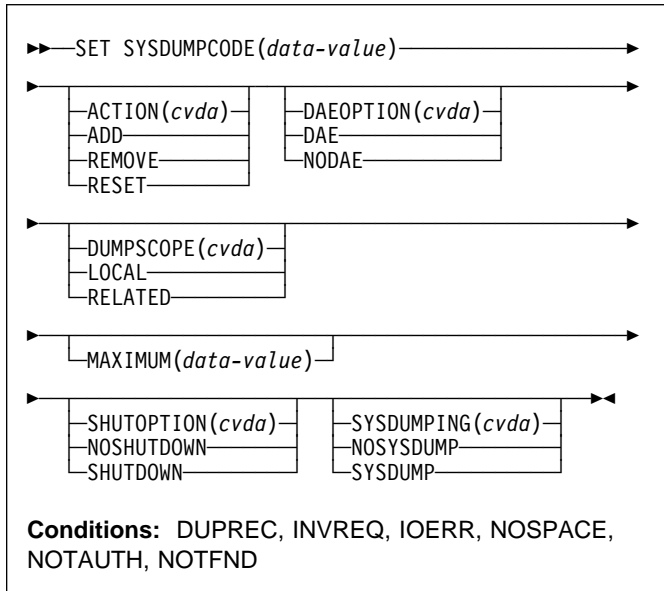
NOTAUTH

RESP2 values:

- + **100** The user associated with the issuing task is not authorized to use this command.

SET SYSDUMPCODE

+ Change an entry in the system dump table.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

The SET SYSDUMPCODE command allows you to change the system dump table entry for a particular dump code, to add a new dump code to the table, or to delete one.

+ The table entry tells CICS what actions to take when a system dump request with this code occurs. Possible actions include taking a system dump (an MVS SDUMP), initiating requests for SDUMPs of related CICS regions, and shutting down CICS. The table entry also indicates how many times this set of actions is to be taken (the MAXIMUM + value); after the maximum is reached, requests are counted + but otherwise ignored.

+ Table updates are recorded in the CICS global catalog and preserved over executions of CICS until a cold start occurs, + except in the case of temporary table entries. CICS creates + a temporary entry when it receives a dump request with a + code for which there is no table entry; these entries, and any + changes to them, last only for the current execution of CICS. + If you want to preserve changes to a temporary entry over + restarts, you need to remove the dump code from the table + and then add it back.

For information about system dumps, see the *CICS/ESA Problem Determination Guide*.

Options

ACTION(cvda)

+ specifies what action is to be taken for the dump code. CVDA values are:

+ ADD An entry for this code is to be added to the table.

REMOVE

+ The entry for this code is to be removed from the table. No other option can be specified on a SET SYSDUMPCODE REMOVE command.

RESET

+ The current number of dump requests for this code is to be set to zero. (See the CURRENT option of the INQUIRE SYSDUMPCODE command.)

DAEOPTION

specifies whether a dump produced for this dump code is eligible for suppression by the MVS Dump Analysis and Elimination (DAE) component. CVDA values are:

DAE The dump is eligible for DAE suppression.

NODAE

The dump is not eligible for DAE suppression—if CICS determines that a dump should be written, MVS will not suppress it. (However, be aware of the SUPPRESS and SUPPRESSALL options in the ADYSETxx parmlib member. These are controlled by the VRADAE and VRANODAE keys in the SDWA. They may lead to dump suppression even though NODAE is set here. For information about DAE, SUPPRESS, and SUPPRESSALL, see the *MVS ESA Planning: Problem Determination and Recovery* manual.

+ When SET SYSDUMPCODE ADD is specified, if you do not also specify DAEOPTION, it will default to NODAE—regardless of the setting of the SIT DAE parameter.

DUMPSCOPE(cvda)

+ specifies whether a request for a dump with this dump code should cause CICS to initiate requests for SDUMPs (system dumps) of “related” CICS regions.

+ A related CICS region is one in the same sysplex, connected by MRO/XCF and doing work on behalf of your CICS region—specifically, a region that has one or more tasks doing work under the same APPC token as a task in your region.

+ This propagation of SDUMP requests occurs only when the table entry for this code also specifies a SYSDUMPING value of SYSDUMP, and only in a sysplex environment executing under MVS/ESA 5.1 and the MVS workload manager.

+ If you specify RELATED in other systems, this will cause an exception condition.

CVDA values are:

- LOCAL
+ SDUMP requests are not to be sent.
- RELATED
+ SDUMP requests are to be sent.
- + LOCAL is the default for entries you add, if you do not
+ specify a DUMPSCOPE value.

MAXIMUM(*data-value*)

- specifies, as a fullword binary value, the maximum
+ number of dumps with this code that CICS should
+ request, in the range 0–999. After the maximum is
+ reached, CICS counts but otherwise ignores dump
+ requests with this code. A value of 999 means there is
+ no limit, and is the default for new entries if you do not
+ specify a MAXIMUM value.

You cannot specify both MAXIMUM and SHUTDOWN.

SHUTOPTION(*cvda*)

- + specifies whether the system is to be shut down after a
+ request for a dump with this dump code. CVDA values
+ are:

NOSHUTDOWN

The system is not to be shut down.

SHUTDOWN

The system is to be shut down.

NOSHUTDOWN is assumed if you omit this value from
+ a SET SYSDUMPCODE ADD command. You cannot
+ specify both MAXIMUM and SHUTDOWN.

SYSDUMPCODE(*data-value*)

- + specifies the 8-character system dump code for which
+ the system dump table entry is to be modified. A valid
+ system dump code contains no leading or imbedded
+ blanks.

SYSDUMPING(*cvda*)

- + specifies whether a system dump request with this code
+ should produce a dump. CVDA values are:

NOSYSDUMP

- + A dump is not to be taken.

SYSDUMP

- + A dump is to be taken.

Even when SYSDUMP is specified, CICS takes a dump
+ only if the number of requests for this code is less than
+ the MAXIMUM and system dumps are not suppressed
+ globally (see the DUMPING option of the INQUIRE
+ SYSTEM command). MVS may also be allowed to
+ suppress the dump if appropriate, depending on the
+ DAEOPTION value.

If the SYSDUMPING option is omitted from a SET
+ SYSDUMPCODE ADD command, SYSDUMP is
+ assumed.

DUPREC

RESP2 values:

- + **10** ADD is specified for a dump code already in the
+ system dump table.

INVREQ

RESP2 values:

- 2** ACTION has an invalid CVDA value.
4 SYSDUMPING has an invalid CVDA value.
+ **5** The MAXIMUM value is out of range.
6 SHUTOPTION has an invalid CVDA value.
+ **7** REMOVE is specified with other options.
+ **8** Both MAXIMUM and SHUTDOWN are specified.
+ **9** The dump code is invalid.
+ **13** DUMPSCOPE has an invalid CVDA value.
+ **14** RELATED requires MVS/ESA 5.1.
+ **15** DAEOPTION has an invalid CVDA value.

IOERR

RESP2 values:

- + **11** An error occurred updating the CICS catalog. The
+ entry is changed for the current run, but is not
+ recorded for restarts.

NOSPACE

RESP2 values:

- + **12** The CICS catalog is full. The entry is changed for
+ the current run, but is not recorded for restarts.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not
+ authorized to use this command.

NOTFND

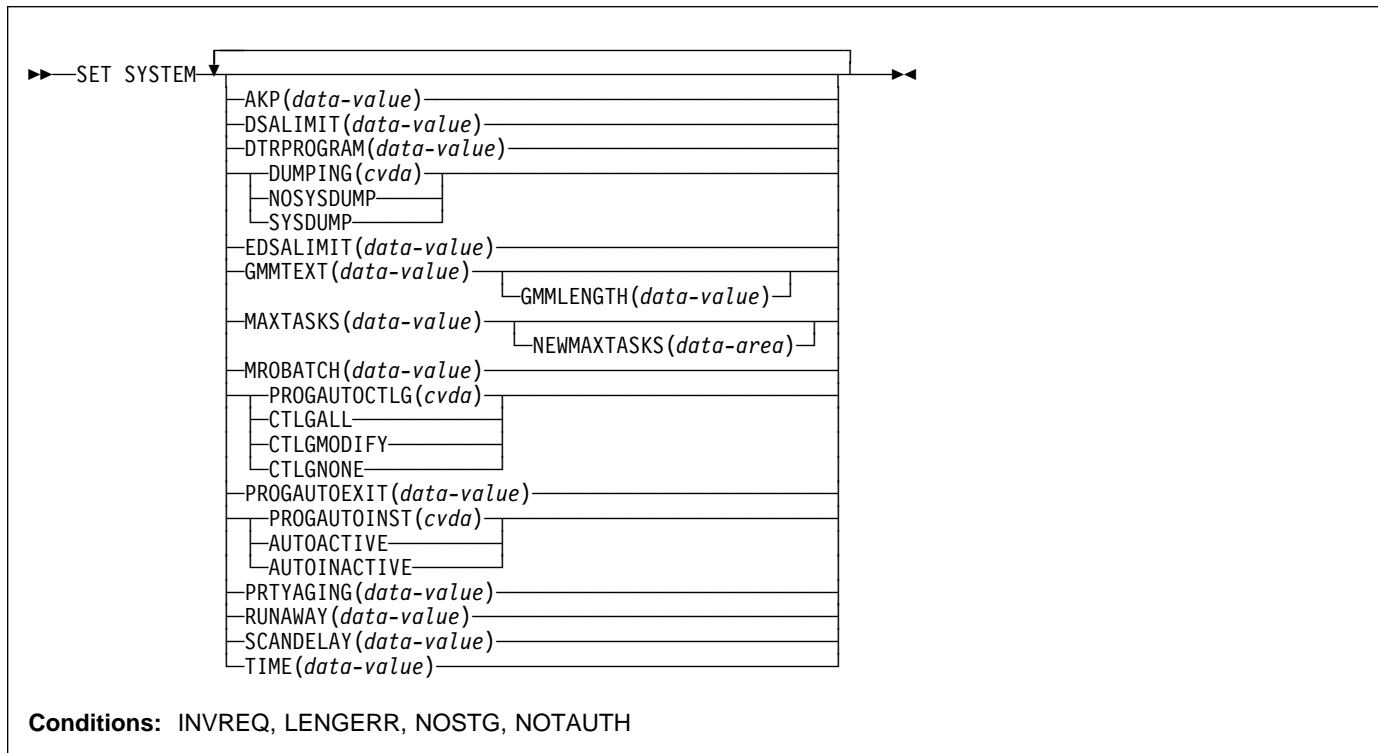
RESP2 values:

- + **1** The dump code cannot be found.

Conditions

SET SYSTEM

+ Change CICS system option values.

**Notes:**

1. For more information about the use of CVDA's, see "CICS-value data areas (CVDA's)" on page 7.
2. The CSCS, ECSCS, USCS, EUSCS, and ERSCS options, each of which returned the size of the storage "cushion" for a particular dynamic storage area, are obsolete in CICS/ESA 4.1. To maintain object compatibility, they are accepted at run-time but ignored. The translator also accepts them, but issues a warning message.

Description

| The SET SYSTEM command allows you to change the | values of some of the options that control the execution of + your CICS system.

+ These values are set initially by system initialization + parameters, described in the *CICS/ESA System Definition + Guide*. System initialization parameters that correspond to + those in this command have the same or similar names, + except where noted, and Table 2 on page 100 lists the + exact correspondence.

Options**AKP(*data-value*)**

specifies, as a fullword binary value, the activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. The number must be either zero, which turns off keypointing, or in the range 200–65535. This value cannot be changed if CICS was initialized without keypointing (that is, with the AKPFREQ system initialization parameter set to zero).

DSALIMIT(*data-value*)

| specifies, as a fullword binary value, the maximum + amount of storage, in bytes, within which CICS can + allocate storage for the four individual dynamic storage + areas (DSAs) that reside below the 16MB boundary. If + DSALIMIT specifies a value lower than the current limit, + CICS may not be able to implement the new limit + immediately, but will attempt to do so over time as + storage is freed. The range for DSALIMIT is + 2MB–16MB.

DTRPROGRAM(*data-value*)

+ specifies the 8-character name of the program that is to + control the dynamic routing of transactions.

DUMPING(*cvda*)

+ specifies a CVDA value indicating whether the taking of CICS system dumps is to be suppressed. CVDA values are:

NOSYSDUMP

+ System dumps are to be suppressed.

SYSDUMP

+ System dumps are not to be suppressed.

| **EDSALIMIT**(*data-value*)

| specifies, as a fullword binary value, the maximum amount of storage, in bytes, within which CICS can allocate storage for the four individual dynamic storage areas that reside above the 16MB boundary. If EDSALIMIT specifies a value lower than the current limit, CICS may not be able to implement the new limit immediately, but will attempt to do so over time as storage is freed. The range for EDSALIMIT is 10MB–2GB.

| **GMMLENGTH**(*data-value*)

| specifies, as a halfword binary value, the length of the “good morning” message text. The range for this value is 1–246.

| **GMMTEXT**(*data-value*)

+ specifies the “good morning” message text, which can be up to 246 characters long.

MAXTASKS(*data-value*)

specifies, as a fullword binary value, the maximum number of tasks that can be eligible for dispatch at any one time in this CICS system. Both active and suspended tasks count toward this limit, but tasks that have not reached the point of initial dispatch do not. System tasks such as terminal and journal control tasks do not count in CICS/ESA 4.1 either, although they did in earlier releases. The value can be in the range 1–999.

MROBATCH(*data-value*)

specifies, as a fullword binary value, the number of events that must occur, from a list of MRO and DASD I/O events on which CICS is waiting, before CICS is posted explicitly to process them. The value must be in the range 1–255.

+ **NEWMAXTASKS**(*data-area*)

+ returns the new value of MAXTASKS, in fullword binary form.

+ When you set MAXTASKS in a SET SYSTEM command, CICS adjusts the value you specify downward if there is not enough storage for the value you request; NEWMAXTASKS tells you what the value is after any such adjustment. CICS also raises the NOSTG condition when it reduces the value, although it continues processing your command.

| **PROGAUTOCTLG**(*cvda*)

| specifies whether and when autoinstalled program definitions are to be cataloged. Cataloged definitions

+ are restored on a warm or emergency restart. Those not cataloged are discarded at shutdown and must be installed again if used in a subsequent execution of CICS.

+ Decisions to catalog are made both at initial installation and whenever an autoinstalled definition is modified, and are based on the PROGAUTOCTLG value at the time. CVDA values are:

| **CTLGALL**

+ Definitions are to be cataloged when installed and when modified.

| **CTLGMODIFY**

+ Definitions are to be cataloged only when modified.

| **CTLGNONE**

+ Definitions are not to be cataloged.

| **PROGAUTOEXIT**(*data-value*)

+ specifies the 8-character name of the user-provided program to be called by the CICS program autoinstall code to provide a model definition.

+ **Note:** This program (and any programs it invokes) must be installed before they can be used in the program autoinstall process, either by explicit PROGRAM definitions or by autoinstall when some other autoinstall program is in force. Otherwise, the program autoinstall process fails when next used, and CICS makes it inactive.

| **PROGAUTOINST**(*cvda*)

+ specifies whether autoinstall for programs is to be active or inactive. When a task requests a program, map set or partition set that is not defined, CICS attempts to create a definition for it automatically if autoinstall for programs is active. If not, CICS raises the PGMIDERR exceptional condition. CVDA values are:

| **AUTOACTIVE**

+ Autoinstall for programs is to be active.

| **AUTOINACTIVE**

+ Autoinstall for programs is to be inactive.

PRTYAGING(*data-value*)

+ specifies, as a fullword binary value, the rate at which CICS is to increase the priority of a task waiting for dispatch. CICS increases the task priority by 1 after each PRTYAGING milliseconds of wait time without a dispatch. The value must be in the range 0–65535.

RUNAWAY(*data-value*)

+ specifies, as a fullword binary value, the default for runaway task time. This value is used for any task executing a transaction with a profile that does not specify runaway task time (see the RUNAWAY option of the INQUIRE TRANSACTION command on page 132).

+ The value must be either zero, which means that runaway task detection is not required for tasks using the default value, or in the range 500–2700000. The

SET SYSTEM

- + value you supply is rounded down to the nearest multiple of 500.

SCANDELAY(*data-value*)

- + specifies, as a fullword binary value, the maximum number of milliseconds between a user task making a terminal I/O request and CICS dispatching the terminal control task to process it. This value is sometimes called the “terminal scan delay,” and is set initially by the system initialization option ICVTSD. The value must be in the range 0–5000.

TIME(*data-value*)

- + specifies, as a fullword binary value, the maximum interval in milliseconds for which CICS gives control to the operating system if no tasks are ready for dispatch. This value is set initially by the ICV system initialization option and is sometimes called the “region exit time interval.” The TIME value must be in the range 100–3600000 and must not be less than the SCANDELAY value. You can determine the current SCANDELAY value, if you are not setting it at the same time, with the INQUIRE SYSTEM SCANDELAY command.

- + **16** CICS reduced the value you requested for MAXTASKS because of storage constraints; see the NEWMAXTASKS option.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not authorized to use this command.

Conditions

INVREQ

RESP2 values:

- + **1** The MAXTASKS value is out of range.
- + **3** The AKP value is out of range.
- | **5** TIME is not in the range 100–3600000.
- + **6** The RUNAWAY value is out of range.
- 7** MROBATCH is not in the range 1–255.
- 9** DUMPING has an invalid CVDA value.
- + **12** AKP was specified, but CICS was initialized without keypointing.
- + **13** TIME is less than SCANDELAY.
- 14** PRTYAGING is not in the range 0–65535.
- 15** SCANDELAY is not in the range 0–5000.
- # **19** The PROGAUTOEXIT value contains an invalid character.
- # **20** DSALIMIT is not in the range 2MB to 16MB.
- | **21** EDSALIMIT is not in the range 10MB to 2GB.
- | **22** There is insufficient MVS storage to allocate DSALIMIT.
- + **23** There is insufficient MVS storage to allocate EDSALIMIT.
- # **24** The DTRPROGRAM value contains an invalid character.
- #

LENGERR

RESP2 values:

- + **20** The GMMLength value is out of range.

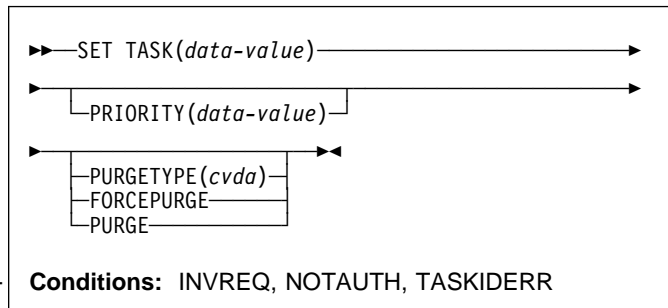
NOSTG

RESP2 values:

SET TASK

Function

- + Purge a task or change its priority.



Description

- + The SET TASK command allows you to purge a task
- + (terminate it abnormally) or to change its priority. Not all
- + tasks can be changed with this command, however; in
- + particular, CICS-created tasks that are essential to system
- + operation are ineligible.

For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Options

PRIORITY(*data-value*)

specifies, as a fullword binary value, the priority you want for the task. The value must be in the range 0–255.

PURGETYPE(*cvda*)

- + specifies that CICS is to purge the task, and indicates conditions for doing so.
- + Purging a task at the wrong time can result in a loss of data integrity or, in some circumstances, can cause CICS to abend. CICS always defers purging until the task reaches a state where the system itself does not appear to be at risk, but you can specify whether CICS also should wait until data integrity can be ensured.
- + If CICS accepts a purge request, it returns a NORMAL response to SET TASK. You can tell whether execution has been deferred by inspecting the RESP2 value. If RESP2 is zero, the purge has been completed; if RESP2 is 13, it has been deferred. CVDA values are:

FORCEPURGE

- + The task is to be terminated as soon as consistent with system integrity and without regard to data integrity.
- + **Note:** CICS cannot always determine whether a forced purge is safe; it is possible to abend the system when you specify FORCEPURGE.

PURGE

- + The task is to be terminated as soon as both system and data integrity can be maintained.
- + **Note:** You cannot purge a task with this CVDA value if the definition of the TRANSACTION it is executing specifies SPURGE=NO.

TASK(*data-value*)

- + specifies the 4-byte packed-decimal sequence number of the task you are changing.

Conditions

INVREQ

RESP2 values:

- 3 PURGETYPE has an invalid CVDA value.
- 4 PRIORITY is not in the range 0–255.
- 5 The task is not in a valid state for purging.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

TASKIDERR

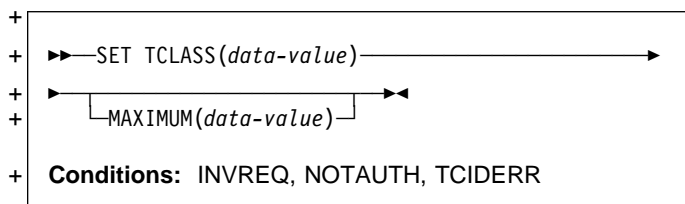
RESP2 values:

- + 1 The task cannot be found.
- + 2 The task is protected by CICS and not eligible for modification with this command.

SET TCLASS

SET TCLASS

+ Set the maximum number of tasks in a transaction class.



+ Description

+ The SET TCLASS command allows you to set the maximum number of tasks in a particular transaction class which are allowed to run concurrently.

+ This command is limited to the numbered classes of earlier releases of CICS and is retained for compatibility with those releases. The SET TRANCLASS command, described on page 194, provides the same function and can be used for either the old numbered or new named classes.

+ Options

+ MAXIMUM(data-value)

+ specifies, as a fullword binary value, the largest number of tasks in the transaction class that are allowed to run concurrently. The value can be in the range 0 through 999. (This value corresponds to the MAXACTIVE value in a SET TRANCLASS command. See the description of this option on page 194 for a description of what happens when you change the MAXACTIVE limit.)

+ TCLASS(data-value)

+ specifies, as a fullword binary value, the number of the task class that you are changing. It must be in the range 0–10.

+ Conditions

+ INVREQ

+ RESP2 values:

+ **2** The MAXIMUM value is not in the range 0-999.

+ NOTAUTH

+ RESP2 values:

+ **100** The user associated with the issuing task is not authorized to use this command.

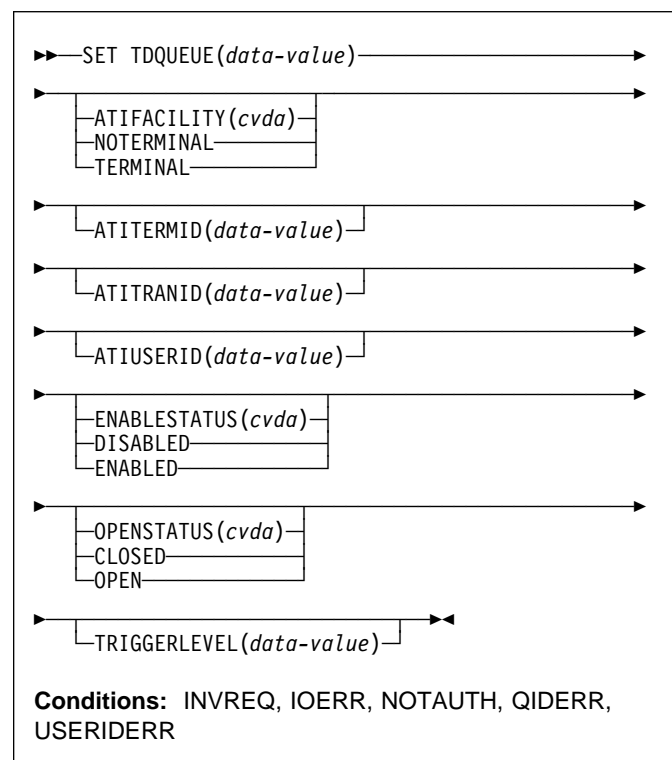
+ TCIDERR

+ RESP2 values:

+ **1** The transaction class cannot be found.

SET TDQUEUE

+ Change the attributes of a transient data queue.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

+ The SET TDQUEUE command allows you to change some attributes of a transient data queue.

+ Transient data queues, also called destinations, are defined in the destination control table (DCT) of CICS. There are two basic types: **intrapartition** and **extrapartition**.

+ Intrapartition queues are managed and stored entirely by CICS, and are eligible for automatic task initiation (ATI), the facility that CICS provides for scheduling tasks automatically. For a transient data queue, ATI is governed by the TRIGGERLEVEL option value. If it is nonzero, CICS automatically creates a task to process the queue when the number of items on the queue reaches this trigger level; a value of zero exempts the queue from ATI.

+ An extrapartition queue is an MVS sequential data set (or a spool file). Extrapartition queues are not subject to ATI.

+ There are two other types of queue: **indirect** and **remote**, both of which point to one of the basic types. You cannot modify the definition of either with a SET TDQUEUE command, however. (See the INQUIRE TDQUEUE command for more information about these queues.)

Options

ATIFACILITY(*cvda*) (intrapartition queues only)

- + specifies whether the queue has a terminal (or session) associated with it. When ATI occurs, this option determines whether the task that CICS creates to process the queue has a principal facility or not. CVDA values are:
- NOTERMINAL
- + ATI tasks are to execute without a principal facility.
- TERMINAL
- + ATI tasks require the terminal specified in ATITERMID as principal facility.

ATITERMID(*data-value*) (intrapartition queues only)

- + specifies the 4-character name of the terminal or session associated with the queue, if any. When CICS creates a task to process the queue, this terminal is the principal facility if the ATIFACILITY value is TERMINAL.
- + You can set this value at any time, but it is used only during ATI, and only when ATI tasks are to have a principal facility. When ATIFACILITY is NOTERMINAL, CICS retains but does not use the ATITERMID value, and does not display it in an INQUIRE TDQUEUE command.

ATITRANID(*data-value*) (intrapartition queues only)

- + specifies the 4-character identifier of the transaction to be executed when CICS initiates a task automatically to process the queue. This value is used only during ATI. CICS does not check the ATITRANID value when you set it but, when ATI occurs, the created task will abend unless the ATITRANID value names a TRANSACTION defined at the time. Furthermore, this transaction must not be defined as remote.

ATIUSERID(*data-value*) (intrapartition queues only)

- + specifies the 8-byte user identifier associated with the queue, if any. If there is no terminal associated with the queue when ATI occurs, CICS assigns this user to the task it creates to process the queue.
- + You can set this value at any time, but it used only during ATI, and only when the ATIFACILITY value is NOTERMINAL. When ATIFACILITY is TERMINAL, CICS retains but does not use the ATIUSERID value, and does not display it in an INQUIRE TDQUEUE command.
- + In addition to the authority checks made for any SET TDQUEUE command, when ATIUSERID is specified, CICS invokes the external security manager to ensure that the user associated with the task issuing the command has authority to act for the user named in ATIUSERID. When the ESM is RACF, this means that the user associated with the task must be defined as a RACF **surrogate** for the user in ATIUSERID.

ENABLESTATUS(*cvda*)

specifies whether the queue can be accessed by applications. CVDA values are:

DISABLED

- + The queue cannot be accessed by applications. You cannot specify this value for a queue whose name begins with the letter "C", because CICS does not allow you to disable CICS-defined queues.

ENABLED

- + The queue can be accessed by applications.
- + For extrapartition queues, changing the ENABLESTATUS value affects only the availability of the queue; CICS does not open or close the associated data set.

OPENSTATUS(*cvda*) (extrapartition queues only)

- + specifies whether the data set associated with the queue is to be open or closed. CVDA values are:
- CLOSED
- + The data set is to be closed.
- OPEN
- + The data set is to be open.
- + You cannot change the OPENSTATUS value of a queue that is DISABLED.

TDQUEUE(*data-value*)

- + specifies the 4-character name of the transient data queue (destination) whose attributes you are changing.

TRIGGERLEVEL(*data-value*) (intrapartition only)

- + specifies, as a fullword binary value, the number of items that must be on the queue for ATI to occur, or, alternatively, that ATI is not to occur. The number must be in the range 0–32767. If is zero, no ATI occurs; otherwise, when the queue reaches the TRIGGERLEVEL depth, CICS creates a task to process it automatically. See also the ATIFACILITY, ATITERMINAL, ATITRANID and ATIUSERID options.

Conditions

INVREQ

RESP2 values:

- 2 TRIGGERLEVEL was specified for an extrapartition queue.
- + 3 The TRIGGERLEVEL value is not in the range 0–32767.
- 4 ATITERMID was specified for an extrapartition queue.
- 5 ATITRANID was specified for an extrapartition queue.
- 6 ATIFACILITY was specified for an extrapartition queue.
- + 7 ATIFACILITY has an invalid CVDA value.
- + 8 OPENSTATUS has an invalid CVDA value.

SET TDQUEUE

- 9 OPENSTATUS was specified for an intrapartition queue.
- + 10 ENABLESTATUS has an invalid CVDA value.
- 11 DISABLED was specified for a queue beginning with "C".
- + 12 The queue is remote.
- + 13 The queue is indirect.
- + 15 OPENSTATUS was specified for a DISABLED queue.
- + 16 OPENSTATUS was specified, but the JCL DDNAME to which the queue definition points was not found.
- + 19 ATIUSERID was specified for an extrapartition queue.
- + 20 The ESM interface is not initialized.
- 21 CICS has received an unknown response from the ESM.
- + 22 The ESM did not respond.

IOERR

RESP2 values:

- + 14 An error occurred opening or closing the data set associated with the queue.
- + 17 The queue cannot be set CLOSED because there is no space in the associated data set.

NOTAUTH

RESP2 values:

- + 23 The user named in the ATIUSERID option is not authorized.
- + 24 The user named in ATIUSERID has been revoked.
- | 25 During SECLABEL processing by the external security manager, an error occurred. For information about security labels, see the *Resource Access Control Facility (RACF) Security Administrator's Guide*.
- | 27 The user named in the ATIUSERID option is not allowed to access the queue.
- 100 The user associated with the issuing task is not authorized to use this command.
- 101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.
- + 102 The user associated with the issuing task is not an authorized surrogate for the user specified in ATIUSERID.

QIDERR

RESP2 values:

- + 1 The queue cannot be found.

USERIDERR

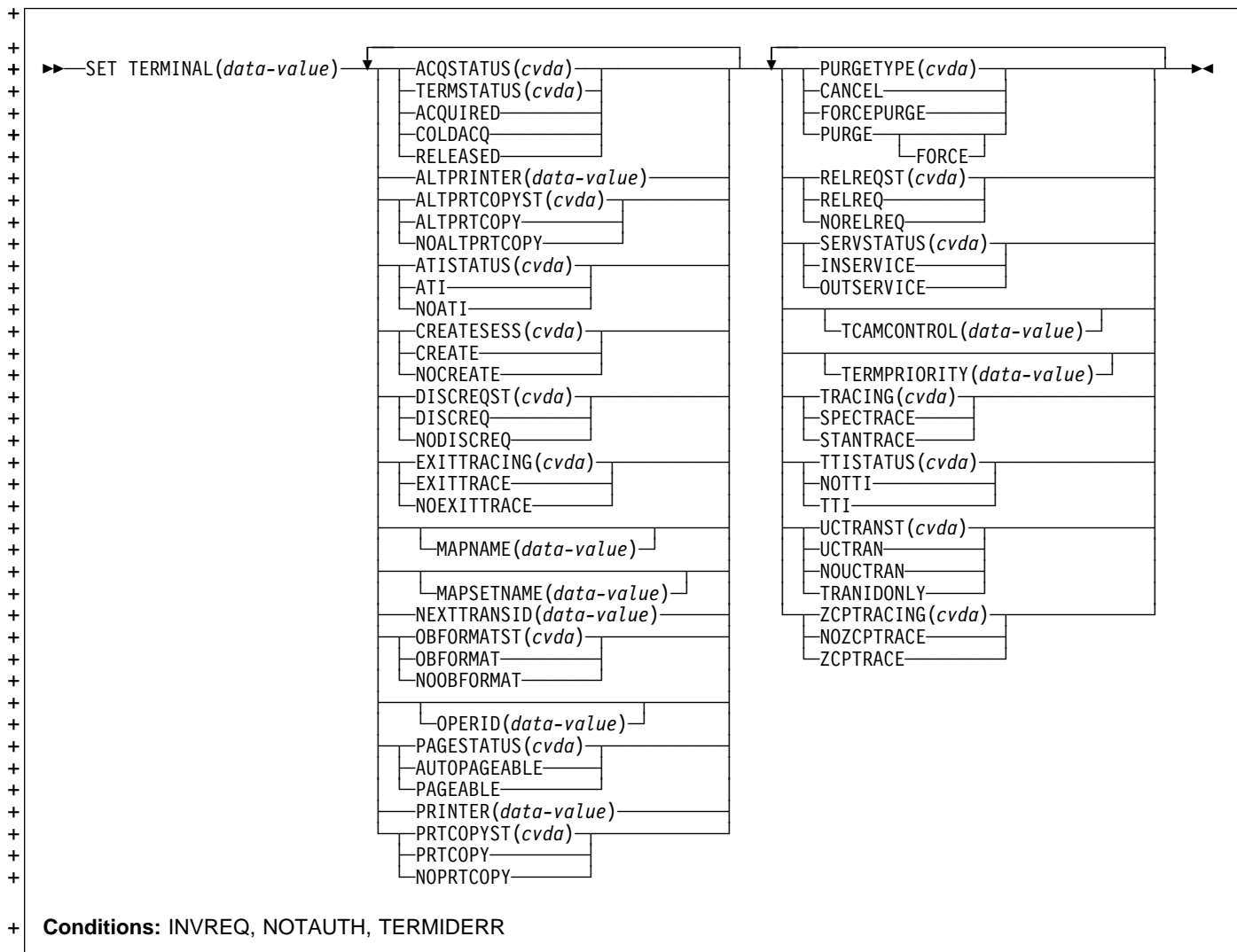
RESP2 values:

- + 28 The user named in ATIUSERID is not known to the ESM.

SET TERMINAL

Change the definition or status of a terminal or cancel work associated with it.

+ **APAR PQ01572**
 + added OPERID to EXEC CICS SET TERMINAL



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

+ The SET TERMINAL command allows you to change both the definition and the current status of a terminal. It applies to physical terminals (those defined with TERMINAL and TYPETERM resource definitions), and many options apply to LUTYPE6.1 sessions as well. You cannot use it for APPC or MRO sessions, except as noted in the description of the PURGETYPE option.

+ In general you cannot use SET TERMINAL for remote terminals either, but the NEXTTRANSID and TRACING options are allowed in some situations (see these options for more information).

+ You also can use SET TERMINAL to purge the task currently executing with the terminal as its principal or alternate facility, and to cancel work scheduled for the terminal. See the PURGETYPE option for details about these functions.

Options

SET TERMINAL

| ACQSTATUS(*cvda*) (VTAM only)

- + specifies the same value as TERMSTATUS and is retained only for compatibility purposes. You should use TERMSTATUS in new applications (see page187).

ALTPRINTER(*data-value*)

- + specifies the 4-character name of the alternate printer for print key requests and ISSUE PRINT commands issued in tasks for which this terminal is the principal facility. This printer is used when the one named in the PRINTER option is not available.
- + If you do not want an alternate designated, set the ALTPRINTER value to blanks. If you do this, you must ensure that the ALTPRTCOPYST value is NOALTPRTCOPY: you can make this change in the same command if necessary.
- + The printer you specify must be a 3270 printer owned by the same CICS system as the terminal, and there are other restrictions. See the ALTPRINTER, ALTPRTCOPY, PRINTER and PRINTCOPY options of a TERMINAL definition in the *CICS/ESA Resource Definition Guide* for full details.

APAR PQ03859
added the following note.

- # **Note:** For VTAM terminals, in a transaction routing environment, this command will not take effect until the next flow across the link from the TOR to the AOR for the named terminal.

ALTPRTCOPYST(*cvda*)

- + specifies whether CICS is to use the hardware copy feature to satisfy a print request for which the alternate printer is used (see the ALTPRINTER option). CVDA values are:

ALTPRTCOPY

- + CICS is to use the hardware copy feature.
- + **Note:** You cannot specify ALTPRTCOPY unless an alternate printer is defined.

NOALTPRTCOPY

CICS is not to use the hardware COPY feature.

ATISTATUS(*cvda*)

- + specifies whether CICS can initiate a task automatically (ATI) with this terminal or session as its principal facility. For sessions, this option also governs whether the session can be allocated as an alternate facility. CVDA values are:
- + ATI ATI and use as an alternate facility are to be allowed.
- + NOATI ATI and use as an alternate facility are not to be allowed. This value cannot be assigned to a terminal whose TTISTATUS value is NOTTI.

CREATESESS(*cvda*) (VTAM only)

- + specifies whether CICS should attempt to acquire the terminal, if necessary, for an automatic task initiation (ATI) request. CVDA values are:

CREATE

The terminal should be acquired.

NOCREATE

- + The terminal should not be acquired.

DISCREQST(*cvda*)

- + specifies whether CICS is to honor a disconnect request from the terminal. A disconnect request results from an ISSUE DISCONNECT command or a CESF (sign-off) task with the GOODNIGHT or LOGOFF option. CVDA values are:

DISCREQ

- + CICS is to honor a request to disconnect this terminal (with a VTAM CLSDST request to terminate the session if the terminal is a VTAM terminal).

NODISCREQ

- + CICS is not to honor a request to disconnect this terminal.

+ EXITTRACING(*cvda*) (VTAM only)

- + specifies whether this terminal is to be traced when CICS is tracing terminal-specific invocations of its VTAM exits (see the TCEXITSTATUS option of the SET TRACEFLAG command). CVDA values are:

EXITTRACE

- + This terminal is to be traced.

NOEXITTRACE

- + This terminal is not to be traced.

+ MAPNAME(*data-value*)

- + specifies the 7-character name of the map which is to be remembered by CICS as the name of the last map sent to this device. If this terminal is a surrogate, the map name specified is returned in the DETACH sequence to the terminal owning system when the currently executing transaction terminates, unless the map name is superseded by a subsequent SEND MAP command. The MAPNAME option can be used to restore a map name which was returned to the application in a previous INQUIRE TERMINAL command. If the terminal is not supported by BMS (for example this terminal is a session), an INVREQ condition is raised with a RESP2 value of 60.

+ MAPSETNAME(*data-value*)

- + specifies the 8-character name of the mapset which is to be remembered by CICS as the name of the last mapset used in a SEND MAP command processed for this terminal. If this terminal is a surrogate, the mapset name specified is returned in the DETACH sequence to the terminal owning system when the currently executing transaction terminates, unless the mapset name is superseded by a subsequent SEND MAP command.

+ The MAPSETNAME option can be used to restore a mapset name which was returned to the application in a previous INQUIRE TERMINAL command. If the terminal is not supported by BMS (for example this terminal is a session), an INVREQ condition is raised with a RESP2 value of 60.

NEXTTRANSID(*data-value*)

+ specifies the 4-character identifier of the transaction to be executed to process the next unsolicited input from this terminal. If you specify blanks, CICS sets the next transaction identifier to nulls to indicate that no such value is defined for the terminal.

+ If no task is executing with the terminal as its principal facility at the time of the SET TERMINAL command, the effect is as if the previous task at the terminal had specified your NEXTTRANSID value in the TRANSID option of its final RETURN command.

+ The same thing happens if there is a task executing, provided the task does not set TRANSID on its final RETURN command and provided it does not abend. Otherwise its TRANSID value (or CICS abend cleanup) overrides your NEXTTRANSID.

+ When there is a task executing (on your CICS system), you can specify NEXTTRANSID for a remote terminal, and the effect is the same. This is the only time you can use NEXTTRANSID for a terminal owned by another region, however; at other times, CICS will raise an INVREQ exception.

+ **Note:** You cannot specify NEXTTRANSID for a terminal defined with a permanent "next transaction" value (see the TRANSACTION option of the INQUIRE TERMINAL command).

OBFORMATST(*cvda*)

+ specifies whether outboard formatting can be used at this terminal. CVDA values are:

NOOBFORMAT

+ Outboard formatting cannot be used.

OBFORMAT

+ Outboard formatting can be used.

Note: OBFORMATST cannot be specified for a console or 3790.

APAR PQ01572

added OPERID to EXEC CICS SET TERMINAL

OPERID(*data-value*)

+ Specifies an operator identification code that is to be associated with the terminal. The identification code can be up to 3 characters long. The operator identification code will continue to be associated with the terminal until it is changed by another SET TERMINAL OPERID command, or until the user signed on at the terminal changes (i.e. until a user signs on or signs off at the terminal).

PAGESTATUS(*cvda*)

+ specifies how pages of BMS messages with a disposition of PAGING are to be delivered to the terminal. CVDA values are:

AUTOPAGEABLE

+ Pages are to be written automatically in sequence.

PAGEABLE

+ Pages are to be written on request from the operator.

PRINTER(*data-value*)

+ specifies the 4-character name of the primary printer for print key requests and ISSUE PRINT commands issued in tasks for which this terminal is the principal facility. This printer is used if available; if not, the printer named in the ALTPRINTER option is second choice.

+ If you want no printer designated for this purpose, set the PRINTER value to blanks. If you do this, you must ensure that the ALTPRINTER value is blanks and that the PRTCOPYST and ALTPRTCOPYST values are NOPRTCOPY and NOALTPRTCOPY respectively; you can make these settings in the same command if necessary.

+ The printer you specify must be a 3270 printer owned by the same CICS as the terminal, and there are other restrictions. See the PRINTER and PRINTCOPY options of the TERMINAL definition in the *CICS/ESA Resource Definition Guide* for full details.

#

APAR PQ03859

#

added the following note.

Note: For VTAM terminals, in a transaction routing environment, this command will not take effect until the next flow across the link from the TOR to the AOR for the named terminal.

PRTCOPYST(*cvda*)

+ specifies whether CICS is to use the hardware copy feature to satisfy a print request for which the primary printer is used (see the PRINTER option). CVDA values are:

NOPRTCOPY

+ CICS is not to use the hardware copy feature.

PRTCOPY

+ CICS is to use the hardware copy feature.

+ **Note:** You cannot specify PRTCOPY unless a primary printer is defined.

PURGETYPE(*cvda*)

+ specifies that work occurring at or scheduled for the terminal or session is to be canceled.

+ The PURGE and FORCEPURGE values allow you to purge (terminate abnormally) the task *currently* using this terminal as its principal or alternate facility. You can use them for VTAM terminals and for LUTYPE6.1 and MRO

SET TERMINAL

+ sessions, but not for non-VTAM terminals or APPC
+ sessions.
+ When there is such a task, these options have the same
+ effect in a SET TERMINAL command as they do in a
+ SET TASK command, except that the RESP2 value for a
+ deferred purge is 53 rather than 13. See page 179 for
+ details on how these options work and precautions about
+ using them. (If there is no task at the terminal, no purge
+ occurs and no exception condition is raised.)

+ The CANCEL option cancels the *scheduled* work
+ represented by the set of automatic initiate descriptors
+ (AIDs) for the terminal (session). It must be used alone,
+ without other options. CANCEL can be used for any
+ type of terminal or any type of session.

+ Each AID represents a request for a task which is to
+ execute with the terminal as its principal facility. An AID
+ can originate from a transient data queue reaching its
+ trigger level, a START command, a ROUTE command,
+ or an internal CICS action. (AIDs represent work
+ scheduled for the current time or earlier. Use the
+ CANCEL command, described in the *CICS/ESA*
+ *Application Programming Reference* manual, to cancel
+ work scheduled in the future, such as an unexpired
+ START or ROUTE command. See also the INQUIRE
+ REQID command on page 91.)

+ **Note:** An AID that results from a transient data queue
+ reaching trigger level ceases to exist as soon as the
+ associated task is initiated, and thus is not subject to a
+ CANCEL issued after task start. However, if the task
+ has not drained its queue by task end, CICS will create
+ another AID to process the queue, whether or not a
+ CANCEL occurred during execution of the task.

+ When a request in one CICS system is associated with
+ a terminal owned by another CICS, both systems
+ maintain an AID to represent the request. If you cancel
+ any such AIDs by issuing a SET TERMINAL CANCEL in
+ the CICS that owns the terminal, that CICS notifies the
+ originating CICS systems to cancel their corresponding
+ AIDs. The originating CICS systems write message
+ DFHTF0100 to their CSMT transient data destinations to
+ indicate how many AIDs were canceled for this reason
+ and how many remain for the terminal. CVDA values
+ are:

CANCEL

+ AIDs associated with this terminal are to be
+ canceled.

+ **Note:** When you specify CANCEL, you cannot
+ use any other options. If your CANCEL request is
+ accepted, you get a NORMAL response. You can
+ tell whether any AIDs were canceled by inspecting
+ the RESP2 value returned; it will be 58 if any AIDs
+ were canceled and 59 if not.

FORCEPURGE

+ The task at the terminal is to be terminated as
+ soon as consistent with system integrity and
+ without regard to data integrity.

PURGE

+ The task at the terminal is to be terminated as
+ soon as both system and data integrity can be
+ maintained.

+ PURGE FORCE is equivalent to FORCEPURGE
+ and is retained only for compatibility. You should
+ use FORCEPURGE in new applications.

+ RELREQST(*cvda*) (VTAM only)

+ specifies whether CICS is to honor requests from VTAM
+ to release the terminal or session. Release occurs on
+ request, unless a task is using the terminal as principal
+ or alternate facility, in which case it is deferred until task
+ end. CVDA values are:

NORELREQ

+ CICS is not to release the terminal.

RELREQ

+ CICS is to release the terminal.

+ SERVSTATUS(*cvda*)

+ specifies whether the terminal is to be available for use.
+ SERVSTATUS corresponds to the INSERVICE option in
+ the TERMINAL definition. CVDA values are:

INSERVICE

+ The terminal is to be available.

OUTSERVICE

+ The terminal is not to be available. If the terminal
+ is the principal or alternate facility of a task at the
+ time you set it to OUTSERVICE, CICS waits until
+ the task completes to make the change, but starts
+ no other tasks at the terminal. (If the task is
+ running under the execution diagnostic facility
+ (EDF), EDF control ends and the task completes
+ without EDF.) Consequently, you need to purge
+ the task with PURGE or FORCEPURGE if you
+ want the change to be immediate when a task is
+ present.

+ When the status of a terminal changes from INSERVICE
+ to OUTSERVICE, CICS signs off the operator and, if it is
+ a VTAM terminal, releases it (sets the TERMSTATUS
+ value to RELEASED). If you change a terminal that is
+ subject to automatic signon from OUTSERVICE to
+ INSERVICE, CICS performs the signon, but it does *not*
+ acquire a released VTAM terminal unless you also
+ specify a TERMSTATUS VALUE of ACQUIRED.

+ **Note:** To prevent unintended effects, CICS does not
+ permit a task to set the status of its own principal facility
+ to OUTSERVICE unless that facility is a printer.

+ TCAMCONTROL(*data-value*) (TCAM only)

+ specifies the current value of the **position** field in the
+ TCAM work area for the terminal. When used, this field
+ indicates whether the work currently being handled is a
+ whole message or the first, an intermediate, or the last
+ segment of a message, and, for input, whether the data
+ contains a record delimiter. The presence or absence
+ of the letter 'C' in the OPTCD value of the DFHTCT

- + TYPE=SDSCI macro for the terminal determines whether
- + the position field is used or not. (See the discussion of
- + TCAM devices in the *CICS/ESA Resource Definition*
- + *Guide*.)

The value must be expressed as a single hexadecimal character. The meanings are:

- + 00 Null (no change)
- 40 Intermediate part of message
- F1 First part of message
- F2 Last part of message
- F3 Whole message
- F4 Intermediate part of message, end of record
- F5 First part of message, end of record
- F6 Last part of message, end of record
- F7 Whole message, end of record.

TERMINAL(*data-value*)

- + specifies the 4-character name of the terminal or session
- + whose definition or status is to be changed.

TERMPRIORITY(*data-value*)

- specifies, as a fullword binary value, the priority of the
- terminal, relative to other terminals, in the range 0–255.
- + The dispatch priority of a task is the smaller of 255 and
- + the sum of the priorities of terminal that is the principal
- + facility, the user signed on at the terminal, and the
- + transaction being executed.

TERMSTATUS(*cvda*) (VTAM only)

- + specifies whether CICS is to be in session with this
- + terminal. CVDA values are:

ACQUIRED

- + CICS is to be in session, and is to acquire a
- + session if none exists. If the SERVSTATUS of the
- + terminal is OUTSERVICE, you must specify
- + INSERVICE as well as ACQUIRED (or
- + COLDACQ), because CICS will not acquire a
- + terminal that is out of service.

COLDACQ

- + COLDACQ requests the same status that
- + ACQUIRED does, and causes any pending
- + resynchronization information to be discarded first.

RELEASED

- + CICS is not to be in session, and is to terminate
- + the session if one exists. Termination occurs
- + immediately unless the terminal is the principal or
- + alternate facility of a task and you have not
- + specified PURGE or FORCEPURGE; in this case
- + CICS waits until the task ends.

TRACING(*cvda*)

- + specifies the type of tracing to be associated with this
- + terminal.
- + For a task that has this terminal as its principal facility,
- + this value is combined with the TRACING option value of

- + the transaction the task is executing to determine
- + whether tracing is standard, special, or suppressed. If
- + the transaction TRACING value is SUPPRESSED, no
- + tracing occurs. Otherwise, tracing is special if either the
- + terminal or the transaction specifies SPECTRACE,
- + standard if both specify STANTRACE. CVDA values
- + are:

SPECTRACE

- + Special tracing is specified.

STANTRACE

- + Standard tracing is specified.

- + **Note:** You can specify TRACING for a remote terminal,
- + but only if there is a task (on your CICS) executing with
- + the terminal as its principal facility. The tracing applies
- + only to your CICS, not to the CICS that owns the
- + terminal, and the change lasts only for the duration of
- + the task (and any successor tasks invoked by the
- + IMMEDIATE option on a final RETURN command).

TTISTATUS(*cvda*)

- + specifies whether this terminal is to be allowed to initiate
- + tasks by sending unsolicited input. CVDA values are:

NOTTI

- + The terminal is not to initiate tasks. You cannot
- + specify this value for a terminal whose
- + ATISTATUS is NOATI. In addition, to prevent
- + unintended effects, CICS does not allow a task to
- + assign this value to its own principal facility unless
- + that facility is a printer.

- + TTI This terminal is to be able to initiate tasks.

UCTRANST(*cvda*)

- + specifies whether input from this terminal is to be
- + translated to upper case automatically, at the time of
- + receipt. CVDA values are:

NOUCTRAN

- + Input from this terminal is not to be translated on
- + receipt. (It still can be translated before
- + presentation to a task if the PROFILE definition for
- + the transaction being executed specifies
- + translation. See the UCTRAN option in the
- + INQUIRE PROFILE command.)

TRANIDONLY

- + This value is the same as NOUCTRAN, with one
- + difference. If the input is unsolicited, and CICS
- + needs to use the initial characters of the input to
- + decide which transaction to execute, that decision
- + is made from a *copy* of the input which has been
- + translated to upper case. There is no difference
- + between these two options in the data presented
- + to a task issuing a RECEIVE.

UCTRAN

- + The input is to be translated on receipt.

SET TERMINAL

+ **APAR PN81733**
+ added this paragraph

+ If a terminal TCTTE is available in a remote system, in
+ either model or surrogate form, a change can be made
+ to TRACING or NEXTTRANSID in the remote definition.
+ This change is not shipped back to the TOR. This allows
+ the user to make a change which applies only to the
+ remote TCTTE. The SET TERMINAL command can
+ also be used to change the UCTRANST option of a
+ surrogate terminal. This change will be shipped back to
+ the TOR and intermediate systems. Attempting to
+ change any other attribute for a model or surrogate
+ terminal will result in INVREQ with RESP2=24.

+ This command may be used to set the upper case
+ translation option for a remote terminal if the named
+ terminal is the principal facility of the task issuing the
+ command. If the remote terminal is not the principal
+ facility the INVREQ condition will be raised with a
+ RESP2 value of 24. The upper case translate option will
+ also be changed in the terminal owning region and any
+ intermediate regions in a daisy chaining setup.

+ **Note:** Translation, whether caused by the UCTRAN
+ value for the terminal or the transaction, can be
+ suppressed with the ASIS option when input is solicited
+ in a conversational task. However, it cannot be
+ suppressed for unsolicited input (the only input in
+ pseudoconversational and non-conversational tasks).

+ ZCPTRACING(*cvda*) (VTAM only)

+ specifies whether this terminal is to be traced as part of
+ CICS tracing for VTAM terminals. CVDA values are:

NOZCPTRACE

+ The terminal is not to be traced.

ZCPTRACE

+ The terminal is to be traced.

Conditions

INVREQ

RESP2 values:

- | 1 TERMSTATUS or ACQSTATUS was specified for
- + a non-VTAM terminal.
- | 2 TERMSTATUS or ACQSTATUS has an invalid
- | CVDA value.
- + 4 ATISTATUS has an invalid CVDA value.
- + 5 The ATISTATUS or TTISTATUS value specified
- + would result in NOATI and NOTTI.
- + 6 CREATESESS was specified for non-VTAM
- + terminal.
- + 7 CREATESESS has an invalid CVDA value.
- + 9 PAGESTATUS has an invalid CVDA value.
- + 11 OUTSERVICE has been specified for the principal
- + facility.
- + 13 SERVSTATUS has an invalid CVDA value.
- + 15 The TERMPRIORITY value is out of range.

- + 17 NOTTI has been specified for the principal facility.
- + 18 TTISTATUS has an invalid CVDA value.
- + 21 PURGETYPE has an invalid CVDA value.
- + 22 TRACING has an invalid CVDA value.
- + 24 An option invalid for a remote terminal was
- + specified.
- + 25 ACQUIRED was specified for a terminal not in
- + service.
- + 26 PURGE was specified, but the task at the terminal
- + is NOTPURGEABLE.
- + 27 EXITTRACING has an invalid CVDA value.
- + 28 ZCPTRACING has an invalid CVDA value.
- + 29 EXITTRACING or ZCPTRACING was specified for
- + a non-VTAM terminal, or VTAM is not installed.
- + 31 An option invalid for a remote terminal was
- + specified, or the terminal, which is remote, is not
- + currently the principal facility of a task on your
- + system.
- + 33 An option other than CANCEL was specified for
- + an APPC session.
- + 34 A permanent transaction has been defined for this
- + terminal (in the TRANSACTION option of the
- + TERMINAL definition).
- + 35 TCAMCONTROL was specified for a non-TCAM
- + terminal.
- + 36 The TCAMCONTROL value is invalid.
- + 37 Automatic signon failed: the terminal was not
- + placed in service.
- + 38 OBFORMATST has an invalid CVDA value.
- + 39 RELREQST has an invalid CVDA value.
- + 40 DISCREQST has an invalid CVDA value.
- + 41 ALTPRTCOPYST has an invalid CVDA value.
- + 42 PRTCOPYST has an invalid CVDA value.
- + 43 UCTRANST has an invalid CVDA value.
- + 44 The ALTPRTCOPYST or ALTPRINTER value
- + would result in ALTPRTCOPY with no alternate
- + printer defined.
- + 45 The PRINTER or ALTPRINTER value would
- + cause an alternate printer to be defined without a
- + primary printer.
- + 46 OBFORMATST was specified for a console or
- + 3790.
- + 48 The PRTCOPYST or PRINTER value would result
- + in PRTCOPY with no primary printer defined.
- + 50 The TERMSTATUS value cannot be honored
- + because VTAM is closed or closing.
- + 51 The PRINTER or ALTPRINTER option was
- + specified for a non-3270 terminal.
- + 52 The PRTCOPYST or ALTPRTCOPST option was
- + specified for a non-3270 terminal.
- + 54 An option other than PURGETYPE was specified
- + for an MRO session.
- + 57 Other options were specified with CANCEL.
- + 60 MAPNAME or MAPSETNAME specified but the
- + terminal is not of a type supported by BMS.

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

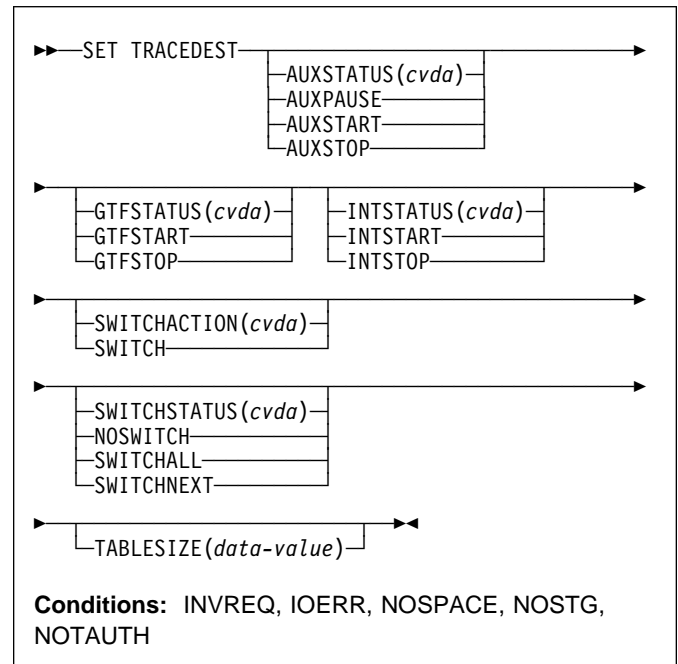
TERMIDERR

RESP2 values:

+ **23** The terminal cannot be found.

SET TRACEDEST

+ Change tracing options.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

+ CICS can write trace entries to three possible destinations:
 + the CICS internal trace table, the auxiliary trace data set, and
 + the MVS Generalized Trace Facility (GTF). The SET TRACEDEST command allows you to specify which destinations are to receive the entries. You also can use it to change the size of the trace table and to switch auxiliary trace data sets.

+ Two other commands, SET TRACEFLAG and SET TRACETYPE, and a CICS-supplied transaction, CETR, can be used to control the number and type of trace entries.

+ Changes made with this command are not recorded in the CICS catalog. Therefore the options affected are always reset to the corresponding system initialization values at CICS startup. These are: INTSTATUS and TRTABSZ (for internal tracing), AUXTR and AUXTRSW (auxiliary tracing), and GTFSTATUS (GTF tracing). See the *CICS/ESA System Definition Guide* for more information about these values.

Options

AUXSTATUS(cvda)

+ specifies whether auxiliary tracing is to occur, that is, whether trace entries are to be written to the active CICS auxiliary trace data set. (See the

SET TRACEDEST

+ SWITCHACTION option for more about auxiliary trace data sets.) CVDA values are:

AUXPAUSE

+ CICS is to stop writing entries, but is to leave the data set open at its current position. A subsequent AUXSTART request will resume writing entries immediately after those that preceded the AUXPAUSE request. You can specify AUXPAUSE only when auxiliary tracing is currently active.

AUXSTART

+ Entries are to be written. The data set is to be opened first if currently closed.

AUXSTOP

+ Entries are not to be written. The data set is to be closed if open. A subsequent AUXSTART request will cause new entries to be written at the start of the data set, overwriting the previous contents, unless there are two auxiliary trace data sets and they are switched between the AUXPAUSE and AUXSTART.

GTFSTATUS(*cvda*)

+ specifies whether trace entries are to be sent to the MVS Generalized Tracing Facility (GTF). CVDA values are:

GTFSTART

+ Entries are to be sent.

GTFSTOP

+ Entries are not to be sent.

+ **Note:** A value of GTFSTART is necessary but not sufficient for recording CICS trace entries on GTF. In addition, CICS must be initialized with GTF support (the GTFTR system initialization option), and GTF must be started in MVS with the TRACE=USR option.

INTSTATUS(*cvda*)

+ specifies whether internal tracing is to occur, that is, whether non-exception trace entries are to be recorded in the internal trace table. (Exception entries are always recorded.) CVDA values are:

INTSTART

+ Entries are to be recorded.

INTSTOP

+ Entries are not to be recorded.

SWITCHACTION(*cvda*)

+ specifies that the auxiliary trace data sets are to be switched.

+ If your system supports auxiliary tracing, it has either one or two auxiliary trace data sets. One is "active," which means it receives trace entries when auxiliary tracing is turned on, and the other, if there are two, is a standby.

+ When there are two, you can reverse their roles by specifying SWITCH. This causes CICS to close the

+ current active data set, open the standby, and reverse the designation of which is active and which standby.

+ If there is only one (or none), SWITCH causes an exception condition, because CICS attempts to open a data set that is not defined.

The CVDA value is:

SWITCH

+ CICS is to perform a switch.

+ **Note:** If you request AUXSTATUS and SWITCHACTION in the same command, AUXSTATUS is set first.

SWITCHSTATUS(*cvda*)

+ specifies what action CICS is to take when the current active auxiliary trace data set fills. When this occurs, CICS cannot continue auxiliary tracing unless a switch or an AUXSTOP-AUXSTART sequence takes place (see the SWITCHACTION and AUXSTATUS options). CVDA values are:

NOSWITCH

+ CICS is to take no action.

SWITCHALL

+ CICS is to switch every time the active data set fills.

SWITCHNEXT

+ CICS is to switch when the current data set is full, but only once; thereafter NOSWITCH is to be in effect.

TABLESIZE(*data-value*)

+ specifies, as a fullword binary value, the size of the internal trace table in kilobytes. If you specify a value that is different from the current trace table size, CICS suspends internal tracing while the change is made, obtains a new table of the requested size, and frees the old one. Data that was in the old table is lost.

+ The table is allocated in multiples of 4KB, with a minimum size of 16KB. Consequently, the value you specify is increased to the next multiple of 4, and to 16 if you specify less than 16. The maximum size is determined by the storage available for the new table.

Conditions

INVREQ

RESP2 values:

- 1 INTSTATUS has an invalid CVDA value.
- 2 A TABLESIZE value of < -1 has been specified.
- 3 AUXSTATUS has an invalid CVDA value.
- 4 SWITCHSTATUS has an invalid CVDA value.
- 5 GTFSTATUS has an invalid CVDA value.
- 6 AUXPAUSE was specified, but auxiliary tracing is not active.
- 11 SWITCHACTION has an invalid CVDA value.

IOERR

RESP2 values:

- 10 A SWITCH request or a SET AUXSTART request resulted in an "open" error for the trace data set.

NOSPACE

RESP2 values:

- 7 There is insufficient space for the new trace table.

NOSTG

RESP2 values:

- + 8 There is insufficient space for an auxiliary trace buffer.
- + 9 There is insufficient space for a GTF trace buffer.

NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not authorized to use this command.

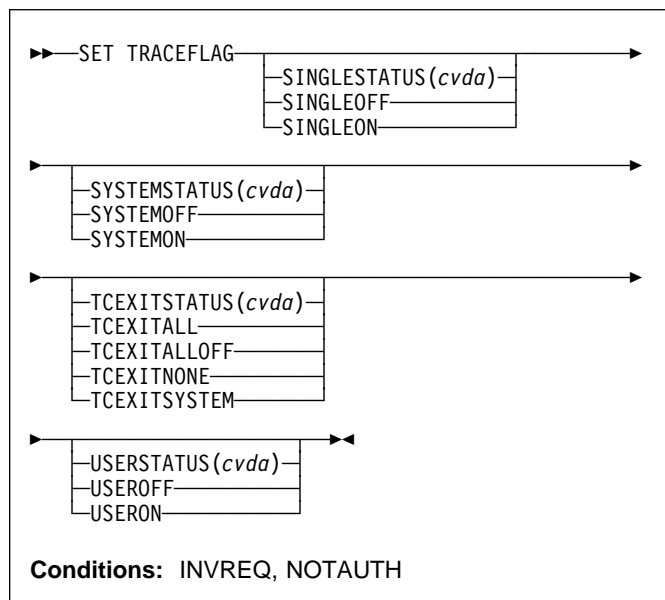
Examples

```
EXEC CICS SET TRACEDEST
        SWITCH
        NOSWITCH
```

- + The SWITCH option tells CICS to switch now from the active auxiliary trace data set (which is not necessarily full) to the + alternate. The NOSWITCH option tells CICS not to switch + when the new active data set fills.

SET TRACEFLAG

Change settings of trace flags.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

- + The SET TRACEFLAG command allows you to change the + flags that control the creation of trace entries in CICS. (See + the *CICS/ESA Problem Determination Guide* for more + information about tracing facilities and control.)
- + Changes made with this command are not recorded in the + CICS catalog, and therefore do not persist beyond CICS + shutdown.

Options

SINGLESTATUS(cvda)

- + specifies whether tracing is to be turned on or
- + suppressed for the task issuing this SET TRACEFLAG
- + command. No nonexception trace entries are made for
- + a task when this flag is off (exception trace entries are
- + *always* recorded).
- + When tracing is allowed, the type of tracing is standard
- + unless special tracing has been requested (in an earlier
- + use of the CETR transaction) for the transaction being
- + executed or the terminal that is the principal facility.
- + CVDA values are:
- SINGLEOFF**
- + Tracing is suppressed.
- SINGLEON**
- + Tracing is allowed.

SET TRACEFLAG

SYSTEMSTATUS(*cvda*)

- + specifies how the system master trace flag is to be set.
- + This flag determines whether CICS makes or
- + suppresses standard trace entries (it does not govern
- + special or exception trace entries). It applies to all tasks
- + and all system activity; however, for standard trace
- + entries to be recorded for any particular task, both the
- + system master flag and the SINGLESTATUS flag for the
- + task must be on. CVDA values are:

SYSTEMOFF

- + Standard tracing is to be suppressed.

SYSTEMON

- + Standard tracing is to be active.

TCEXITSTATUS(*cvda*) (VTAM only)

- + specifies which invocations of the CICS VTAM exits are
- + to be traced.
- + Two types of exit activity can be traced: invocations
- + associated with particular terminals that have been
- + designated for VTAM exit tracing ("terminal-specific"
- + activity), and invocations not related to any particular
- + terminal ("nonterminal specific" activity). You can trace
- + both types or nonterminal specific activity only.

CVDA values are:

TCEXITALL

- + All exit activity is to be traced.

TCEXITALLOFF

- + Terminal-specific activity is not to be traced. The
- + status of nonterminal specific tracing is to remain
- + unchanged.

TCEXITNONE

- + No exit activity is to be traced.

TCEXITSYSTEM

- + Non-terminal specific activity is to be traced, but
- + terminal-specific activity is not.

USERSTATUS(*cvda*)

- + specifies whether the user master trace flag is to be set
- + on or off. This flag governs whether non-exception user
- + trace entries are recorded or suppressed (entries that
- + specify the EXCEPTION option are never suppressed).
- + It applies to all tasks; however, for user entries to be
- + recorded for any particular task, both the user master
- + trace flag and the SINGLESTATUS flag for that task
- + must be on. CVDA values are:

USEROFF

- + User tracing is suppressed.

USERON

- + User tracing is allowed.

Conditions

INVREQ

RESP2 values:

- 1 SYSTEMSTATUS has an invalid CVDA value.
- 2 USERSTATUS has an invalid CVDA value.
- 3 SINGLESTATUS has an invalid CVDA value.
- 4 TCEXITSTATUS has an invalid CVDA value.
- + 5 TCEXITSTATUS is specified but VTAM is not
- + installed.

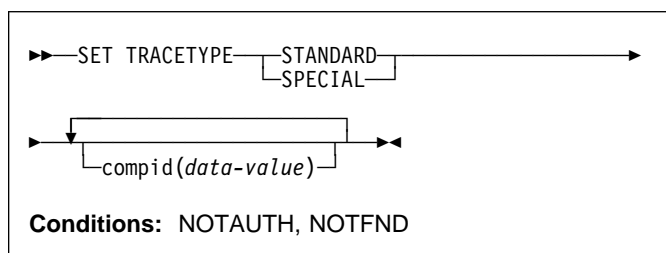
NOTAUTH

RESP2 values:

- 100 The user associated with the issuing task is not
- authorized to use this command.

SET TRACETYPE

+ Change the tracing levels of CICS components.



Description

The SET TRACETYPE command allows you to change the + levels of tracing for one or more CICS components.

+ Each CICS component has trace levels defined separately + for standard CICS tracing and special tracing (see the + *CICS/ESA Problem Determination Guide* for definitions of + these terms and for information about CICS tracing in + general). You can set either type for any number of + components in a SET TRACETYPE command, but you can + set only one type per command.

+ For each component that you specify, you define the trace + levels as a bit string. The bits are read from left to right; that + is, the first bit corresponds to trace level 1, the second to + trace level 2, and so on. A value of 1 turns on the trace + level; 0 turns it off. For example, X'C0000000' turns trace + levels 1 and 2 on and all others off.

+ Although most components define only a few trace levels, + you must provide a 32-bit (4-byte) data value. CICS ignores + bits that do not correspond to trace levels, and thus it does + not matter whether you specify 0 or 1 for them.

Options

compid(data-value)

+ sets the trace levels for the CICS component identified + by "compid," using the bits in the data value as + described above.

+ CICS components can be identified by a 2-character + designation or, in some cases, a descriptive keyword. + For example, to set the trace levels for the storage + manager component of CICS, you can specify either:

+ SET TRACETYPE SM(data-value)

+ or

+ SET TRACETYPE STORAGE(data-value)

The following list shows all the 2-character identifiers, and the keywords for those components that have them.

AP	APPLICATION	Application
BF		Built-in functions
BM		Basic mapping support

CP	CPI	Common programming interface
DC		Dump control
DD	DIRMGR	Directory manager
DI		Batch data interchange
DM	DOMAINMGR	Domain manager
DS	DISPATCHER	Dispatch manager
DU	DUMP	Dump manager
EI		EXEC interface
FC		File control and DL/I
GC	GLOBALCATLG	CICS global catalog manager
IC		Interval control
IS		Intersystem communication
JC		Journal control
KC		Task control
KE	KERNEL	Kernel
LC	LOCALCATLG	CICS local catalog manager
LD	LOADER	Program load manager
LM	LOCKMGR	Lock manager
ME	MESSAGE	Message manager
MN	MONITOR	Monitoring manager
PA	PARAMGR	Parameter manager
PC		Program control
PG	PROGMGR	Program manager
SC		Storage control
SM	STORAGE	Storage manager
SP		Syncpoint manager
ST	STATISTICS	Statistics manager
SZ		Front-end programming interface
TC		Terminal control
TD		Transient data
TI	TIMER	Timer manager
TR	TRACE	Trace manager
TS		Temporary storage
UE		User exit interface
US	USER	User interface
XM	TRANMGR	Transaction manager
XS	SECURITY	Security manager

SPECIAL

+ specifies that you want to set levels for special tracing + for the components listed.

STANDARD

+ specifies that you want to set levels for standard tracing + for the components listed.

Conditions

NOTAUTH

RESP2 values:

100 The user associated with the issuing task is not authorized to use this command.

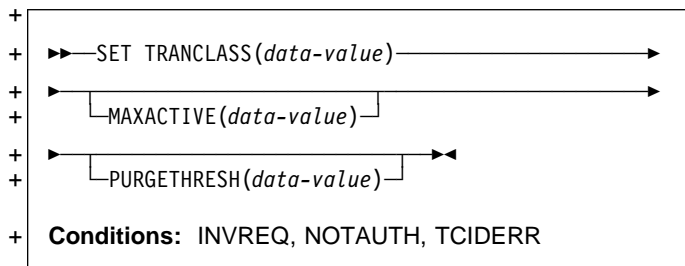
NOTFND

RESP2 values:

+ **1** At least one CICS component was not accessible. + Trace levels were set for the other components.

SET TRANCLASS

+ Set limits for a transaction class.



+ Description

+ The SET TRANCLASS command allows you to change the limits that govern tasks within a particular transaction class. These are the maximum number of tasks that can run concurrently (the MAXACTIVE value) and the maximum number that can queue awaiting initial dispatch (the PURGETHRESH value).

+ Options

+ MAXACTIVE(data-value)

+ specifies, as a fullword binary value, the largest number of tasks in the transaction class that can run concurrently. The value can be in the range 0–999. Raising the MAXACTIVE limit has an immediate effect if the old value of MAXACTIVE has caused queuing, because CICS dispatches queued tasks up to the new MAXACTIVE value. The effect of lowering MAXACTIVE, however, is gradual. Tasks in the class that are already running are allowed to complete normally, but new tasks are not dispatched until the number running drops below the new limit. If you lower MAXACTIVE to zero, you prevent any task in the class from starting execution until MAXACTIVE is increased.

+ PURGETHRESH(data-value)

+ specifies, as a fullword binary value, one more than the maximum number of tasks in this class that can be queued awaiting initial dispatch. Queuing can occur either because the number of active tasks in the class is already at the MAXACTIVE value or because the maximum for the system has been reached (see the MAXTASKS option in the INQUIRE SYSTEM command). Tasks that arrive while the queue is at its PURGETHRESH limit are purged (abended with a code of AKCC). The PURGETHRESH value for a class can be between 0–1000000. A value of zero means there is no purge threshold limit, that is, that any number of tasks can be queued. A value of one means that no tasks can be queued.

+ Raising the PURGETHRESH limit allows more transactions to queue and has an effect only when a task is attached that would have been purged if the old value were in effect.

+ However, if you lower the PURGETHRESH limit beyond the current size of the queue, enough queued tasks are abended to reduce the queue to the new limit. If you raise MAXACTIVE at the same time you lower PURGETHRESH, CICS dispatches as many queued tasks as possible before purging queued tasks, to minimize the number of tasks that get abended. Tasks are abended in priority order, starting with the lowest priority task.

+ TRANCLASS(data-value)

+ specifies the 8-character name of the transaction class that you are changing. If the class is one of the numbered classes used in earlier releases of CICS, its name is DFHTCLnn, where “nn” is the two-digit class number.

+ Conditions

+ INVREQ

+ RESP2 values:

- + 2 The MAXACTIVE value is not in the range 0–999.
- + 3 The PURGETHRESH value is not in the range 0–1000000.

+ NOTAUTH

+ RESP2 values:

- + 100 The user associated with the issuing task is not authorized to use this command.

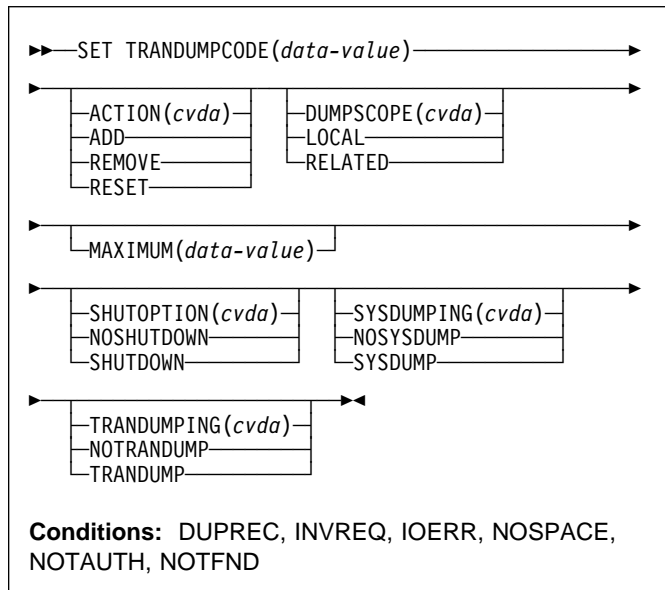
+ TCIDERR

+ RESP2 values:

- + 1 The transaction class cannot be found.

SET TRANDUMPCODE

- + Change an entry in the transaction dump table.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

- + The SET TRANDUMPCODE command allows you to change the transaction dump table entry for a particular dump code, to add a new dump code to the table, or to delete one.
- + The table entry tells CICS what actions to take when a transaction dump request with this code is received.
- + Possible actions include taking a transaction dump, taking a system dump (an MVS SDUMP), initiating requests for SDUMPs of related CICS regions, and shutting down CICS.
- + The table entry also indicates how many times this set of actions is to be taken (the MAXIMUM value); after the maximum is reached, requests are counted but otherwise ignored.
- + Table updates are recorded in the CICS global catalog and preserved over executions of CICS until a cold start occurs, except in the case of temporary table entries. CICS creates a temporary entry when it receives a dump request with a code for which there is no table entry; these entries, and any changes to them, last only for the current execution of CICS.
- + If you want preserve changes to a temporary entry over restarts, you need to remove the dump code from the table and then add it back.
- + For information about transaction dumps, see the *CICS/ESA Problem Determination Guide*.

Options

ACTION(cvda)

- + specifies what action is to be taken for the dump code. CVDA values are:
 - + ADD An entry for this code is to be added to the table.
 - + REMOVE The entry for this code is to be removed from the table. No other options can be specified on a REMOVE request.
 - + RESET The current number of dump requests for this dump code is to be set to zero. (See the CURRENT option of the INQUIRE TRANDUMPCODE command.)

DUMPSCOPE(cvda)

- + specifies whether a request for a dump with this dump code should cause CICS to initiate requests for SDUMPs (system dumps) of “related” CICS regions.
- + A related CICS region is one in the same sysplex, connected by MRO/XCF and doing work on behalf of the task that caused the dump request—specifically, a region that has a task doing work under the same APPC token as this task.
- + This propagation of SDUMP requests occurs only when the table entry for this code also specifies a SYSDUMPING value of SYSDDUMP, and only in a sysplex environment executing under MVS/ESA 5.1 and the MVS workload manager. In other systems, specifying RELATED will cause an exception condition.

CVDA values are:

LOCAL

- + SDUMP requests are not to be sent.

RELATED

- + SDUMP requests are to be sent.

- + LOCAL is the default for entries you add, if you do not specify a DUMPSCOPE value.

MAXIMUM(data-value)

- + specifies, as a fullword binary value, the maximum number of times CICS should take the set of actions indicated in the dump table entry. After the maximum is reached, CICS counts but otherwise ignores dump requests with this code. The valid range is 0–999. A value of 999 means there is no limit, and is the default used if you omit this option from an ADD request.

You cannot specify both MAXIMUM and SHUTDOWN on a SET TRANDUMPCODE command.

SHUTOPTION(cvda)

- + specifies whether the CICS system is to be shut down after a request for a dump with this dump code. CVDA values are:

SET TRANDUMPCODE

NOSHUTDOWN

The system is not to be shut down.

SHUTDOWN

The system is to be shut down.

If this option is omitted from an ADD request, NOSHUTDOWN is assumed.

You cannot specify both MAXIMUM and SHUTDOWN on a SET TRANDUMPCODE command.

SYSDUMPING(*cvda*)

- + specifies whether a system dump (an MVS SDUMP)
- + should be taken when a transaction dump request with
- + this code is received. CVDA values are:

NOSYSDDUMP

- + A system dump is not to be taken.

SYSDUMP

- + A system dump is to be taken.

- + Even when SYSDUMP is specified, CICS takes a dump
- + only if the number of requests for this code is less than
- + the MAXIMUM and system dumps are not suppressed
- + globally (see the DUMPING option of the INQUIRE
- + SYSTEM command).

If this option is omitted from an ADD request, NOSYSDDUMP is assumed.

TRANDUMPCODE(*data-value*)

- + specifies the 4-character transaction dump code for
- + which the transaction dump table entry is to be changed.
- A valid transaction dump code has no leading or
- imbedded blanks.

TRANDUMPING(*cvda*)

- + specifies whether a transaction dump should be taken
- + when a transaction dump request with this code is
- + received. CVDA values are:

NOTRANDUMP

- + A transaction dump is not to be taken.

TRANDUMP

- + A transaction dump is to be taken.

Even when TRANDUMP is specified, CICS will dump only when the count of requests for this code is no greater than the MAXIMUM.

If this option is omitted from an ADD request, TRANDUMP is assumed.

Conditions

DUPREC

RESP2 values:

- + **10** ADD is specified for a dump code already in the
- + transaction dump table.

INVREQ

RESP2 values:

- 2** ACTION has an invalid CVDA value.
- 3** TRANDUMPING has an invalid CVDA value.
- 4** SYSDUMPING has an invalid CVDA value.
- + **5** The MAXIMUM value is out of range.
- 6** SHUTOPTION has an invalid CVDA value.
- + **7** REMOVE is specified with other options.
- + **8** Both MAXIMUM and SHUTDOWN are specified.
- + **9** The dump code is invalid.
- 13** DUMPSCOPE has an invalid CVDA value.
- + **14** RELATED requires MVS/ESA 5.1.

IOERR

RESP2 values:

- + **11** An error occurred updating the CICS catalog. The
- + entry is changed for the current run, but is not
- + recorded for restarts.

NOSPACE

RESP2 values:

- + **12** The CICS catalog is full. The entry is changed for
- + the current run, but is not recorded for restarts.

NOTAUTH

RESP2 values:

- 100** The user associated with the issuing task is not
- authorized to use this command.

NOTFND

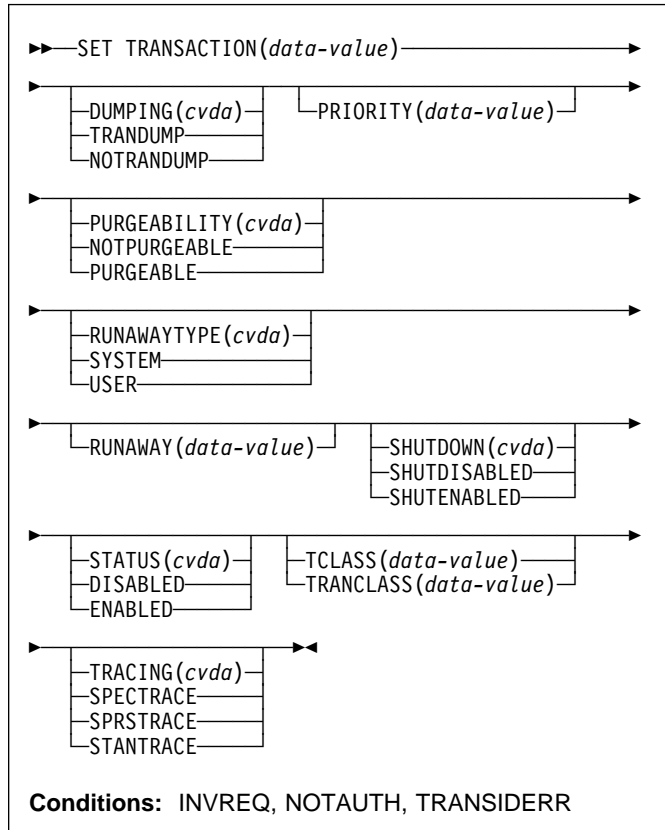
RESP2 values:

- + **1** The dump code cannot be found.

SET TRANSACTION

Function

- + Change a TRANSACTION definition.



For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” on page 7.

Description

- + The SET TRANSACTION command allows you to change some attributes of a TRANSACTION definition.
- + You can change only the definitions in the local CICS system with this command. If you change a transaction that executes remotely (that is, one that specifies a REMOTESYSTEM value), your changes are made, but they have no effect on the definition in the remote system to which the local definition points, and therefore no effect on tasks that execute the transaction.
- + Changing a TRANSACTION definition affects only future tasks; to change a task already executing the transaction, use the SET TASK command.

Options

DUMPING(*cvda*)

- + specifies whether CICS should take a transaction dump if a task executing this transaction terminates abnormally. CVDA values are:

NOTRANDUMP

- + No dump should be taken.

TRANDUMP

- + A dump should be taken.

- + This value applies only toabend dumps and has no effect on DUMP TRANSACTION commands.

PRIORITY(*data-value*)

- + specifies, as a fullword binary value, the priority of this transaction relative to other transactions in the CICS system. The value must be in the range 1–255.

PURGEABILITY(*cvda*)

- + returns a CVDA value indicating whether CICS is allowed to purge this task (that is, to terminate it abnormally). Purge requests come from SET TASK PURGE commands (or CEMT equivalents), and CICS can generate them internally to reclaim resources to relieve a system stall condition. CVDA values are:

NOTPURGEABLE

The task cannot be purged.

PURGEABLE

The task can be purged.

The PURGEABILITY value is set initially by the SPURGE option in the definition of the TRANSACTION this task is executing.

+ RUNAWAY(*data-value*)

- + specifies, as a fullword binary value, the “runaway task” time, in milliseconds, for tasks executing this transaction. The value must be in the range 0–2700000. If a task keeps control of the processor for more than this interval, CICS assumes it is in a loop and abends it. If the value is zero, CICS does not monitor the task for a runaway condition.

- + **Note:** If you specify RUNAWAY, you must set RUNAWAYTYPE to USER in the same SET command, even if RUNAWAYTYPE already has a value of USER.

RUNAWAYTYPE(*cvda*)

- + specifies where the runaway task time for a task executing this transaction should be obtained. CVDA values are:

SYSTEM

- + The system default for runaway-task time should be used. (An INQUIRE SYSTEM command with the RUNAWAY option will tell you what the system value is.)

SET TRANSACTION

USER

- + The RUNAWAY value for this transaction should
- + be used. You must specify a value for RUNAWAY
- + when you specify USER.

SHUTDOWN(*cvda*)

- + specifies whether this transaction can be executed
- + during CICS shutdown by a task created to process
- + unsolicited terminal input (the transaction also can be
- + executed in this situation if it appears in the transaction
- + list table (XLT) for shutdown). CVDA values are:

SHUTDISABLED

The transaction cannot be executed.

SHUTENABLED

The transaction can be executed.

STATUS(*cvda*)

- + specifies whether the transaction is to be available for
- + use. CVDA values are:

DISABLED

The transaction is not available for use.

ENABLED

The transaction is available for use.

Transactions beginning with the letter "C" are CICS-supplied and cannot be disabled.

TCLASS(*data-value*)

- + specifies, as a fullword binary value, the transaction
- + class to which the transaction is to belong. When
- + executed under CICS/ESA 4.1, SET TRANSACTION
- + TCLASS sets the TRANCLASS value in a
- + TRANSACTION definition.
- + TCLASS is provided only for compatibility with earlier
- + releases of CICS, where transaction classes were
- + numbered rather than named, and you can use it only to
- + assign a name of the form DFHTCLnn, where nn is the
- + # number you specify, in the range 00–10. (It does not
- + change the TCLASS value in the TRANSACTION
- + definition, which CICS maintains for situations in which
- + the same TRANSACTION definition is used for several
- + different releases. See the descriptions of TCLASS and
- + TRANCLASS in the INQUIRE TRANSACTION command
- + for more information.)

TRACING(*cvda*)

- + specifies the type of tracing to be done for tasks
- + executing this transaction. See the *CICS/ESA Problem*
- + *Determination Guide* for definitions of tracing types.
- + CVDA values are:

SPECTRACE

- + Tracing is to be special.

SPRSTRACE

- + Tracing is to be suppressed.

STANTRACE

- + Tracing is to be standard.

TRANCLASS(*data-value*)

- + specifies the 8-character name of the transaction class
- + to which this transaction is to belong.

TRANSACTION(*data-value*)

- + specifies the 4-character name of the TRANSACTION
- + definition that you are changing.

Conditions

INVREQ

RESP2 values:

- | 2 PURGEABILITY has an invalid CVDA value.
- | 3 STATUS has an invalid CVDA value.
- + 4 DISABLED has been specified for a
- + CICS-supplied transaction.
- | 5 The TCLASS or TRANCLASS name is not known.
- | 7 TRACING has an invalid CVDA value.
- | 8 DUMPING has an invalid CVDA value.
- + 9 The PRIORITY value is out of range.
- | 10 RUNAWAYTYPE has an invalid CVDA value.
- | 11 SHUTDOWN has an invalid CVDA value.
- + 12 USER has been specified without a RUNAWAY
- + value.
- + 13 RUNAWAY has been specified without a
- + RUNAWAYTYPE value of USER.
- + 14 The RUNAWAY value is out of range.

NOTAUTH

RESP2 values:

- | 100 The user associated with the issuing task is not
- | authorized to use this command.
- | 101 The user associated with the issuing task is not
- | authorized to access this particular resource in the
- | way required by this command.

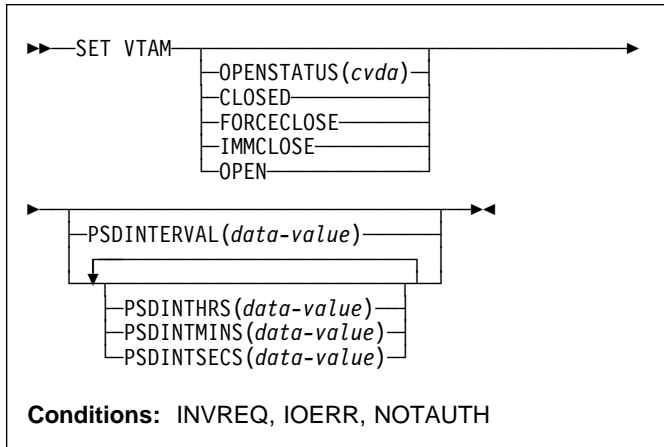
TRANSIDERR

RESP2 values:

- | 1 The transaction cannot be found.

SET VTAM

+ Modify the CICS VTAM connection.



For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" on page 7.

Description

+ The SET VTAM command allows you to establish or terminate the CICS connection to VTAM and to modify the persistent session delay interval value that CICS passes to VTAM.

Options

OPENSTATUS(cvda)

+ specifies whether or not CICS is to have a connection to VTAM (that is, whether the VTAM ACB is to be open or closed) and, if CICS must close the ACB to comply, how the shut down should be done. CVDA values are:

CLOSED

+ The connection is to be closed. If it is currently open, CICS is to quiesce all VTAM activity and then close the VTAM ACB. Tasks using VTAM terminals or sessions are allowed to complete before closure, but new tasks requiring VTAM are not begun.

FORCECLOSE

+ The connection is to be closed. If currently open, CICS is to close the VTAM ACB immediately. Both VTAM sessions and tasks using VTAM will terminate abnormally as a result.

APAR PQ13599

added this paragraph

If VTAM persistent sessions support is in use (that is, the SIT parameter PSDINT has a value other than 0) the sessions will persist when the ACB is closed. They will be unbound if the ACB is

re-opened dynamically. If CICS has shutdown in any way other than a normal shutdown (see the *CICS/ESA Recovery and Restart Guide* for a description of normal shutdown), and is restarted with the system initialization START option START=AUTO, then an emergency restart will be performed, and the sessions will be restored.

IMMCLOSE

+ The connection is to be closed. If currently open, CICS is to terminate abnormally any tasks using VTAM immediately, do an orderly shutdown of all its VTAM sessions, and then close the VTAM ACB.

OPEN

A connection is to be open. If the VTAM ACB is closed, CICS is to open it.

APAR PQ01573

added this paragraph

+ If CICS is using VTAM multi-node persistent sessions, and VTAM has been restarted after an abend, opening the VTAM ACB causes CICS to restore the persistent sessions that VTAM has retained. However, CICS does not restore APPC synclevel 2 sessions, which are unbound.

PSDINTERVAL(data-value)

+ specifies the persistent session delay (PSD) interval value, which determines whether and for how long VTAM is to hold sessions in recovery-pending state after a CICS failure. The range for the value is 0–23:59:59. Zero causes the persistent session feature not to be used; sessions are terminated at the time of the failure.

+ **Note:** Zero is the only value allowed in a system which is eligible for the extended recovery facility; see the XRFSTATUS option in the INQUIRE SYSTEM command.

+ When you specify a PSD interval, CICS sets the system initialization option PSDINT (see the *CICS/ESA System Definition Guide* for more about this option). CICS passes this value to VTAM whenever it opens the ACB. This will occur immediately if you specify an OPENSTATUS value of OPEN in the same SET SYSTEM command, or if the VTAM ACB is already open and you do not close it. If the ACB is closed or being closed, or if the open attempt fails, the new value is established on the next successful open.

+ If VTAM is below the level that supports persistent sessions (Version 3.4.1), VTAM rejects the PSD request. CICS then sets the PSDINT system option value to zero and returns an INVREQ condition, but goes on to continue any other processing you requested. The INVREQ occurs when the value is passed to VTAM, which may be later than the command that set it, as explained above. Consequently, you can get this condition on a command that does not specify a PSD interval.

- + The PSD interval can be expressed in several ways:
- + • A 4-byte packed decimal composite, in the format
- + “**0hhmss+**”, using the PSDINTERVAL option.
- + • With separate hours, minutes, and seconds, using
- + the PSDINTHRS, PSDINTMINS, and PSDINTSECS
- + options. You can use these options singly or in any
- + combination.

+ When you use PSDINTERVAL or more than one of the
 + separate options, the minutes and seconds portions of
 + the value must not be greater than 59 (PSDINTMINS or
 + PSDINTSECS used alone can exceed 59). For
 + example, you could express an interval of 1 hour and 30
 + minutes in any of the following ways:

- + • PSDINTERVAL(13000)
- + • PSDINTHRS(1), PSDINTMINS(30)
- + • PSDINTMINS(90)
- + • PSDINTSECS(5400)

| **PSDINTHRS**(*data-value*)

+ specifies the hours component of the PSD interval, in
 + fullword binary form (see the PSDINTERVAL option).

| **PSDINTMINS**(*data-value*)

+ specifies the minutes component of the PSD interval, in
 + fullword binary form (see the PSDINTERVAL option).

| **PSDINTSECS**(*data-value*)

+ specifies the seconds component of the PSD interval, in
 + fullword binary form (see the PSDINTERVAL option).

| Conditions

| INVREQ

| RESP2 values:

- | **1** VTAM is not present in the system.
- | **2** OPENSTATUS has an invalid CVDA value.
- + **4** The PSDINTERVAL value is out of range.
- + **5** The PSDINTHRS value is out of range.
- + **6** The PSDINTMINS value is out of range.
- + **7** The PSDINTSECS value is out of range.
- + **8** A PSDINTERVAL value > 0 was specified in an
 + XRF-eligible system.
- + **9** VTAM reported an error while an attempt was
 + being made to set the PSD interval.
- + **10** A PSD interval has been specified but either the
 + VTAM currently in use (or the VTAM library used
 + when the terminal control table was assembled)
 + does not support persistent sessions. The interval
 + may have been specified earlier than this
 + command; see the PSDINTERVAL description. If
 + OPEN was also requested, CICS has opened the
 + VTAM ACB.
- + **11** The ACB has opened successfully, but an error
 + occurred in at least one of the sessions that
 + persisted from the previous failure.
- + **12** Your OPEN request did not complete because
 + another task subsequently requested a close of
 + the VTAM connection.

- | **13** An error occurred during recovery of sessions, and
 + the VTAM ACB will be closed as a result.
- + **14** CICS is performing cleanup processing following a
 + predatory XRF takeover. CICS rejects OPEN
 + requests with this error, without invoking VTAM,
 + during this activity. OPEN requests are processed
 + as usual as soon as cleanup is complete.

SET VTAM

| **IOERR**

| RESP2 values:

| **n** An error occurred during the opening of the ACB.
+ If CICS could not process the request, the RESP2
+ value is 3. If VTAM detected the failure, CICS
+ returns the VTAM FDBK2 code in RESP2: you
+ can look up these errors in the *VTAM*
+ *Programming* manual, under ACB OPEN and
+ CLOSE return codes.

| **NOTAUTH**

| RESP2 values:

| **100** The user associated with the issuing task is not
| authorized to use this command.

Appendix A. CICS-value data areas used by all commands

This appendix lists the CICS-value data-area (CVDA) values and their numeric equivalents for all of the EXEC CICS commands. CVDA's are described beginning on page 7.

Using the tables

This appendix consists of three tables. The first, beginning below, is in character sequence of the CVDA values. The second, beginning on page 207, is in numeric sequence. The third, beginning on page 211, gives the CVDA values returned by the INQUIRE TERMINAL|NETNAME DEVICE command.

CVDA's and numeric values in alphabetic sequence

CVDA	Value
ACQFAIL	515
ACQUIRED	69
ACQUIRING	71
ACTIVE	181
ADD	291
ADDABLE	41
ADDFAIL	519
ADVANCE	265
ALARM	501
ALLCONN	169
ALLOCATED	81
ALLQUERY	431
ALTERABLE	52
ALTERNATE	197
ALTPRTCOPY	446
ANY	158
APLKYBD	391
APLTEXT	393
APPC	124
APPCPARALLEL	374
APPCSINGLE	373
APPLICATION	559
ASACTL	224
ASCII7	616
ASCII8	617
ASSEMBLER	150
ATI	75
ATTENTION	524
AUDALARM	395
AUTOACTIVE	630
AUTOARCH	262
AUTOCONN	170
AUTOINACTIVE	631
AUTOPAGEABLE	80
AUTOSTART	618
AUXILIARY	247
AUXPAUSE	313
AUXSTART	312
AUXSTOP	314
AVAILABLE	95
BACKOUT	192
BACKTRANS	397
BASE	10
BASESPACE	664
BDAM	2
BEGINSESSION	510
BELOW	159
BGAM	63
BLK	47
BLOCKED	16
BROWSABLE	39
BSAM	61
BTAM	62
BUSY	612
C	149
CANCEL	526
CANCELLED	624
CD	491
CEDF	370
CICS	660
CICSDATAKEY	379
CICSEXECKEY	381
CICSTABLE	101
CLEAR	640
CLOSED	19
CLOSELEAVE	261
CLOSEREQUEST	22
CLOSING	21
CMDPROT	673
CMDSECNO	205
CMDSECYES	207
COBOL	151
COBOLII	375
COLDACQ	72
COLDQUERY	433
COLDSTART	266
COLOR	399
COMMIT	208
CONFFREE	82
CONFRECEIVE	83
CONFSEND	84
CONNECTED	690
CONSOLE	66
CONTROLSHUT	623
CONVERSE	600
CONVIDLE	518
COPY	401
CREATE	67
CTLGALL	632
CTLGMODIFY	633
CTLGNONE	634
CTRLABLE	56
CURRENT	260
DAE	684
DATA	508
DATASTREAM	543
DEC	46
DEFAULT	198
DEFRESP1	497
DEFRESP1OR2	528
DEFRESP2	498
DEFRESP3	499
DELAY	637
DELETABLE	43
DELETEFAIL	520

CVDA	Value	CVDA	Value
DEREGERROR	679	INACTIVE	378
DEREGISTERED	678	INBOUND	547
DEST	235	INDIRECT	122
DISABLED	24	INITCOMPLETE	628
DISABLING	25	INOUT	532
DISCARDFAIL	513	INPUT	226
DISCREQ	444	INSERVICE	73
DISK1	252	INSTALLED	550
DISK2	253	INSTALLFAIL	512
DISK2PAUSE	254	INTRA	222
DISPATCHABLE	228	INTSTART	310
DPLSUBSET	383	INTSTOP	311
DS3270	615	INVALID	359
DUALCASE	403	IRC	121
DYNAMIC	178	ISOLATE	658
EB	490	KATAKANA	415
EMERGENCY	268	KEYED	8
EMPTY	210	KSDS	6
EMPTYREQ	31	LE370	377
ENABLED	23	LIC	493
ESDS	5	LIGHTPEN	417
EVENT	334	LOCAL	605
EXCEPT	332	LOG	54
EXCEPTRESP	523	LOGICAL	216
EXCI	650	LOGTERM	269
EXCTL	48	LOSE	544
EXITTRACE	362	LPA	165
EXTENDEDDES	405	LUP	541
EXTRA	221	LUSTAT	525
EXTSECURITY	194	LUW	246
FAILEDDBKOUT	357	LU61	125
FAILINGBKOUT	358	MAIN	248
FCLOSE	273	MAP	155
FINALQUIESCE	183	MAPSET	155
FINPUT	270	MCHCTL	241
FIRSTINIT	625	MDT	506
FIRSTQUIESCE	182	MODEL	370
FIXED	12	MORE	492
FMH	502	MSRCONTROL	419
FMHPARM	385	NEGATIVE	530
FOPEN	272	NEWCOPY	167
FORCE	342	NEWSSESSION	485
FORCECLOSE	351	NOALARM	500
FORCECLOSING	353	NOALTPRTCOPY	447
FORCEPURGE	237	NOAPLKYBD	392
FORMATEDF	606	NOAPLTEXT	394
FORMATTED	542	NOATI	76
FORMFEED	407	NOAUDALARM	396
FOUTPUT	271	NOAUTOARCH	263
FREE	85	NOBACKTRANS	398
FREEING	94	NOCEDF	371
FULL	212	NOCLEAR	641
FULLAPI	384	NOCMDPROT	674
FWDRECOVABLE	354	NOCOLOR	400
GENERIC	651	NOCONV	556
GMT	604	NOCONVERSE	601
GOINGOUT	172	NOCOPY	402
GTFSTART	317	NOCREATE	68
GTFSTOP	318	NOCTL	223
HEX	45	NODAE	685
HFORM	409	NODISCREQ	445
HILIGHT	413	NODUALCASE	404
HOLD	163	NOEMPTYREQ	32
IGNORE	1	NOEVENT	335
IMMCLOSE	350	NOEXCEPT	333
IMMCLOSING	352	NOEXCTL	49

CVDA	Value	CVDA	Value
NOEXITTRACE	363	NOTREADY	259
NOEXTENDEDDES	406	NOTRECOVABLE	30
NOFMH	503	NOTREQUIRED	667
NOFMHPARM	386	NOTSOS	669
NOFORMATEDF	607	NOTTABLE	100
NOFORMFEED	408	NOTTI	78
NOHFORM	410	NOTUPDATABLE	38
NOHIGHLIGHT	414	NOUCTRAN	451
NOHOLD	164	NOVALIDATION	441
NOISOLATE	657	NOVFORM	412
NOKATAKANA	416	NOWAIT	341
NOLIGHTPEN	418	NOWRITE	275
NOLOG	55	NOZCPTRACE	365
NOMDT	507	OBFORMAT	421
NOMSGJRNL	531	OBOPERID	387
NOMSRCONTROL	420	OBTAINING	96
NONAUTOCONN	171	OFF	200
NONCICS	661	OK	274
NONE	496	OLD	26
NOOBFORMAT	422	OLDCOPY	162
NOOBOPERID	388	OLDSESSION	486
NOOUTLINE	424	ON	201
NOPARTITIONS	426	OPEN	18
NOPERF	331	OPENING	20
NOPRESETSEC	243	OPENOUTPUT	257
NOPRINTADAPT	428	OUTLINE	423
NOPROGSYMBOL	430	OUTPUT	227
NOPRTCOPY	449	OUTSERVICE	74
NOQUERY	432	PAGEABLE	79
NOREENTPROT	681	PARTITIONS	425
NORELREQ	443	PARTITIONSET	156
NORMALBKOUT	356	PATH	11
NORMALRESP	522	PENDBEGIN	558
NOSECURITY	196	PENDDATA	560
NOSHUTDOWN	289	PENDFREE	86
NOSOSI	435	PENDING	126
NOSTSN	487	PENDPASS	565
NOSWITCH	285	PENDRECEIVE	87
NOSYSCONNECT	654	PENDRELEASE	562
NOSYNCPPOINT	603	PENDSTART	561
NOSYSDUMP	185	PENDSTSN	557
NOTADDABLE	42	PENDUNSOL	564
NOTALTERABLE	53	PERF	330
NOTAPPLIC	1	PHASEIN	168
NOTASKSTART	608	PHYSICAL	215
NOTBROWSABLE	40	PLI	152
NOTBUSY	613	PL1	152
NOTCDEB	495	POSITIVE	529
NOTCONNECTED	691	POST	636
NOTCTRLABLE	57	PRESETSEC	242
NOTDEFINED	659	PRIMARY	110
NOTDELETABLE	44	PRINTADAPT	427
NOTEMPTY	211	PRIVATE	174
NOTERMINAL	214	PROGRAM	154
NOTEXTKYBD	437	PROGSYMBOL	429
NOTEXTPRINT	439	PROTECTED	504
NOTFWDRCVBLE	361	PRTCOPY	448
NOTINBOUND	546	PURGE	236
NOTINIT	376	PURGEABLE	160
NOTINSTALLED	551	READABLE	35
NOTKEYED	9	READBACK	209
NOTLPA	166	READONLY	275
NOTPENDING	127	READY	258
NOTPURGEABLE	161	RECEIVE	88
NOTRANDUMP	187	RECOVERABLE	29
NOTREADABLE	36	RECOVERED	277

CVDA	Value	CVDA	Value
REENTPROT	680	SYSDUMP	184
REGERROR	677	SYSTEM	643
REGISTERED	670	SYSTEMOFF	320
RELATED	675	SYSTEMON	319
RELEASE	563	TAKEOVER	111
RELEASED	70	TAPE1	250
RELEASING	549	TAPE2	251
RELREQ	442	TASK	233
REMOTE	4	TASKSTART	611
REMOVE	276	TCAM	64
# REMTABLE	103	TCAMSNA	65
REQUIRED	666	TCEXITALL	366
RESET	290	TCEXITALLOFF	369
# RESSECINT	203	TCEXITNONE	368
RESSECNO	202	TCEXITSYSTEM	367
RESSECYES	204	TERM	234
REVERTED	264	TERMINAL	213
ROLLBACK	89	TEXTKYBD	436
ROUTE	638	TEXTPRINT	438
RRDS	7	THIRDDINIT	627
RTR	527	TIMEOUT	511
RU	494	TPS55M2	552
RUNNING	229	TPS55M3	553
SCS	614	TPS55M4	554
SECONDINIT	626	TPS55M5	555
SEND	90	TRANDUMP	186
SESSION	372	TRANIDONLY	452
SESSIONFAIL	517	TTI	77
SESSIONLOST	516	T3278M2	533
SETFAIL	514	T3278M3	534
SHARE	27	T3278M4	535
SHARED	173	T3278M5	536
SHUTDOWN	288	T3279M2	537
SHUTDISABLED	645	T3279M3	538
SHUTENABLED	644	T3279M4	539
SIGNEDOFF	245	T3279M5	540
SIGNEDON	244	UCTRAN	450
SINGLEOFF	324	UNAVAILABLE	672
SINGLEON	323	UNBLOCKED	17
SMF	255	UNDEFINED	14
SOS	668	UNDETERMINED	355
SOSI	434	UNENABLED	33
SPECIFIC	652	UNPROTECTED	505
SPECTRACE	177	UNREGISTERED	671
SPRSTRACE	175	UNSOLDATA	521
STANDBY	629	UPDATABLE	37
STANTRACE	176	USER	642
START	635	USERDATAKEY	380
STARTED	609	USEREXECKEY	382
STARTUP	180	USEROFF	322
STATIC	179	USERON	321
STOPPED	610	USERTABLE	102
STSN	509	VALID	360
STSNSET	488	VALIDATION	440
STSNTEST	489	VARIABLE	13
SUBSPACE	663	VFORM	411
SURROGATE	371	VSAM	3
SUSPENDED	231	VTAM	60
SWITCH	188	WAIT	340
SWITCHALL	287	WARMSTART	267
SWITCHING	225	WIN	545
SWITCHNEXT	286	XCF	665
SYNCFREE	91	XM	123
SYNCRECEIVE	92	XNOTDONE	144
SYNCSEND	93	XOK	143
SYSCONNECT	653	ZCPTRACE	364

CVDA and numeric values in numeric sequence

Value	CVDA
1	NOTAPPLIC
1	IGNORE
2	BDAM
3	VSAM
4	REMOTE
5	ESDS
6	KSDS
7	RRDS
8	KEYED
9	NOTKEYED
10	BASE
11	PATH
12	FIXED
13	VARIABLE
14	UNDEFINED
16	BLOCKED
17	UNBLOCKED
18	OPEN
19	CLOSED
20	OPENING
21	CLOSING
22	CLOSEREQUEST
23	ENABLED
24	DISABLED
25	DISABLING
26	OLD
27	SHARE
29	RECOVERABLE
30	NOTRECOVERABLE
31	EMPTYREQ
32	NOEMPTYREQ
33	UNENABLED
35	READABLE
36	NOTREADABLE
37	UPDATABLE
38	NOTUPDATABLE
39	BROWSABLE
40	NOTBROWSABLE
41	ADDABLE
42	NOTADDABLE
43	DELETABLE
44	NOTDELETABLE
45	HEX
46	DEC
47	BLK
48	EXCTL
49	NOEXCTL
52	ALTERABLE
53	NOTALTERABLE
54	LOG
55	NOLOG
56	CTRLABLE
57	NOTCTRLABLE
60	VTAM
61	BSAM
62	BTAM
63	BGAM
64	TCAM
65	TCAMSNA
66	CONSOLE
67	CREATE
68	NOCREATE

Value	CVDA
69	ACQUIRED
70	RELEASED
71	ACQUIRING
72	COLDACQ
73	INSERVICE
74	OUTSERVICE
75	ATI
76	NOATI
77	TTI
78	NOTTI
79	PAGEABLE
80	AUTOPAGEABLE
81	ALLOCATED
82	CONFFREE
83	CONFRECEIVE
84	CONFSEND
85	FREE
86	PENDFREE
87	PENDRECEIVE
88	RECEIVE
89	ROLLBACK
90	SEND
91	SYNCFREE
92	SYNCRECEIVE
93	SYNCSEND
94	FREEING
95	AVAILABLE
96	OBTAINING
100	NOTTABLE
101	CICSTABLE
102	USERTABLE
# 103	REMTABLE
110	PRIMARY
111	TAKEOVER
121	IRC
122	INDIRECT
123	XM
124	APPC
125	LU61
126	PENDING
127	NOTPENDING
143	XOK
144	XNOTDONE
149	C
150	ASSEMBLER
151	COBOL
152	PLI
152	PL1
154	PROGRAM
155	MAP
156	PARTITIONSET
158	ANY
159	BELOW
160	PURGEABLE
161	NOTPURGEABLE
162	OLDCOPY
163	HOLD
164	NOHOLD
165	LPA
166	NOTLPA
167	NEWCOPY
168	PHASEIN
169	ALLCONN
170	AUTOCONN
171	NONAUTOCONN
172	GOINGOUT

Value	CVDA	Value	CVDA
173	SHARED	257	OPENOUTPUT
174	PRIVATE	258	READY
175	SPRSTRACE	259	NOTREADY
176	STANTRACE	260	CURRENT
177	SPECTRACE	261	CLOSELEAVE
178	DYNAMIC	262	AUTOARCH
179	STATIC	263	NOAUTOARCH
180	STARTUP	264	REVERTED
181	ACTIVE	265	ADVANCE
182	FIRSTQUIESCE	266	COLDSTART
183	FINALQUIESCE	267	WARMSTART
184	SYSDUMP	268	EMERGENCY
185	NOSYSDUMP	269	LOGTERM
186	TRANDUMP	270	FINPUT
187	NOTRANDUMP	271	FOUTPUT
188	SWITCH	272	FOPEN
192	BACKOUT	273	FCLOSE
194	EXTSECURITY	274	OK
196	NOSECURITY	275	READONLY
197	ALTERNATE	276	REMOVE
198	DEFAULT	277	RECOVERED
200	OFF	285	NOSWITCH
201	ON	286	SWITCHNEXT
202	RESSECNO	287	SWITCHALL
# 203	RESSECINT	288	SHUTDOWN
204	RESSECYES	289	NOSHUTDOWN
205	CMDSECNO	290	RESET
207	CMDSECYES	291	ADD
208	COMMIT	310	INTSTART
209	READBACK	311	INTSTOP
210	EMPTY	312	AUXSTART
211	NOTEMPTY	313	AUXPAUSE
212	FULL	314	AUXSTOP
213	TERMINAL	317	GTFSTART
214	NOTTERMINAL	318	GTFSTOP
215	PHYSICAL	319	SYSTEMON
216	LOGICAL	320	SYSTEMOFF
221	EXTRA	321	USERON
222	INTRA	322	USEROFF
223	NOCTL	323	SINGLEON
224	ASACTL	324	SINGLEOFF
225	SWITCHING	330	PERF
226	INPUT	331	NOPERF
227	OUTPUT	332	EXCEPT
228	DISPATCHABLE	333	NOEXCEPT
229	RUNNING	334	EVENT
231	SUSPENDED	335	NOEVENT
233	TASK	340	WAIT
234	TERM	341	NOWAIT
235	DEST	342	FORCE
236	PURGE	350	IMMCLOSE
237	FORCEPURGE	351	FORCECLOSE
241	MCHCTL	352	IMMCLOSING
242	PRESETSEC	353	FORCECLOSING
243	NOPRESETSEC	354	FWDRECOVABLE
244	SIGNEDON	355	UNDETERMINED
245	SIGNEDOFF	356	NORMALBKOUT
246	LUW	357	FAILEDGBKOUT
247	AUXILIARY	358	FAILINGBKOUT
248	MAIN	359	INVALID
250	TAPE1	360	VALID
251	TAPE2	361	NOTFWDRCVBLE
252	DISK1	362	EXITTRACE
253	DISK2	363	NOEXITTRACE
254	DISK2PAUSE	364	ZCPTRACE
255	SMF	365	NOZCPTRACE

Value	CVDA	Value	CVDA
366	TCEXITALL	432	NOQUERY
367	TCEXITSYSTEM	433	COLDQUERY
368	TCEXITNONE	434	SOSI
369	TCEXITALLOFF	435	NOSOSI
370	MODEL	436	TEXTKYBD
370	CEDF	437	NOTEXTKYBD
371	SURROGATE	438	TEXTPRINT
371	NOCEDF	439	NOTEXTPRINT
372	SESSION	440	VALIDATION
373	APPC SINGLE	441	NOVALIDATION
374	APPCPARALLEL	442	RELREQ
375	COBOLII	443	NORELREQ
376	NOTINIT	444	DISCREQ
377	LE370	445	NODISCREQ
378	INACTIVE	446	ALTPRTCOPY
379	CICSDATAKEY	447	NOALTPRTCOPY
380	USERDATAKEY	448	PRTCOPY
381	CICSEXECKEY	449	NOPRTCOPY
382	USEREXECKEY	450	UCTRAN
383	DPLSUBSET	451	NOUCTRAN
384	FULLAPI	452	TRANIDONLY
385	FMHPARM	485	NEWSESSION
386	NOFMHPARM	486	OLDSESSION
387	OOPERID	487	NOSTSN
388	NOOOPERID	488	STSNSET
391	APLKYBD	489	STSNTEST
392	NOAPLKYBD	490	EB
393	APLTEXT	491	CD
394	NOAPLTEXT	492	MORE
395	AUDALARM	493	LIC
396	NOAUDALARM	494	RU
397	BACKTRANS	495	NOTCDEB
398	NOBACKTRANS	496	NONE
399	COLOR	497	DEFRESP1
400	NOCOLOR	498	DEFRESP2
401	COPY	499	DEFRESP3
402	NOCOPY	500	NOALARM
403	DUALCASE	501	ALARM
404	NODUALCASE	502	FMH
405	EXTENDED DS	503	NOFMH
406	NOEXTENDED DS	504	PROTECTED
407	FORMFEED	505	UNPROTECTED
408	NOFORMFEED	506	MDT
409	HFORM	507	NOMDT
410	NOHFORM	508	DATA
411	VFORM	509	STSN
412	NOVFORM	510	BEGINSESSION
413	HILIGHT	511	TIMEOUT
414	NOHILIGHT	512	INSTALLFAIL
415	KATAKANA	513	DISCARDFAIL
416	NOKATAKANA	514	SETFAIL
417	LIGHTPEN	515	ACQFAIL
418	NOLIGHTPEN	516	SESSIONLOST
419	MSRCONTROL	517	SESSIONFAIL
420	NOMSRCONTROL	518	CONVIDLE
421	OBFORMAT	519	ADDFAIL
422	NOOBFORMAT	520	DELETFAIL
423	OUTLINE	521	UNSOLDATA
424	NOOUTLINE	522	NORMALRESP
425	PARTITIONS	523	EXCEPTRESP
426	NOPARTITIONS	524	ATTENTION
427	PRINTADAPT	525	LUSTAT
428	NOPRINTADAPT	526	CANCEL
429	PROGSYMBOL	527	RTR
430	NOPROGSYMBOL	528	DEFRESP1OR2
431	ALLQUERY	529	POSITIVE

Value	CVDA	Value	CVDA
530	NEGATIVE	635	START
531	NOMSGJRNL	636	POST
532	INOUT	637	DELAY
533	T3278M2	638	ROUTE
534	T3278M3	640	CLEAR
535	T3278M4	641	NOCLEAR
536	T3278M5	642	USER
537	T3279M2	643	SYSTEM
538	T3279M3	644	SHUTENABLED
539	T3279M4	645	SHUTDISABLED
540	T3279M5	650	EXCI
541	LUP	651	GENERIC
542	FORMATTED	652	SPECIFIC
543	DATASTREAM	653	SYSCONNECT
544	LOSE	654	NOSYSCONNECT
545	WIN	657	NOISOLATE
546	NOTINBOUND	658	ISOLATE
547	INBOUND	659	NOTDEFINED
549	RELEASING	660	CICS
550	INSTALLED	661	NONCICS
551	NOTINSTALLED	663	SUBSPACE
552	TPS55M2	664	BASESPACE
553	TPS55M3	665	XCF
554	TPS55M4	666	REQUIRED
555	TPS55M5	667	NOTREQUIRED
556	NOCONV	668	SOS
557	PENDSTSN	669	NOTSOS
558	PENDBEGIN	670	REGISTERED
559	APPLICATION	671	UNREGISTERED
560	PENDDATA	672	UNAVAILABLE
561	PENDSTART	675	RELATED
562	PENDRELEASE	677	REGERROR
563	RELEASE	678	DEREGISTERED
564	PENDUNSOL	679	DEREGERROR
565	PENDPASS	684	DAE
600	CONVERSE	685	NODAE
601	NOCONVERSE	690	CONNECTED
602	SYNCPOINT	691	NOTCONNECTED
603	NOSYNCPOINT		
604	GMT		
605	LOCAL		
606	FORMATEDF		
607	NOFORMATEDF		
608	NOTASKSTART		
609	STARTED		
610	STOPPED		
611	TASKSTART		
612	BUSY		
613	NOTBUSY		
614	SCS		
615	DS3270		
616	ASCII7		
617	ASCII8		
618	AUTOSTART		
623	CONTROLSHUT		
624	CANCELLED		
625	FIRSTINIT		
626	SECONDINIT		
627	THIRDINIT		
628	INITCOMPLETE		
629	STANDBY		
630	AUTOACTIVE		
631	AUTOINACTIVE		
632	CTLGALL		
633	CTLGMODIFY		
634	CTLGNONE		

+ CVDA values for the DEVICE option

CVDA sequence	
BATCHLU	191
BIPROG	160
BISYNCH	128
CDRDLPRT	24
CONTNLU	189
HARDCOPY	32
INTACTLU	190
ISCMCONV	209
LUCMODGRP	210
LUCSESS	211
LUTYPE4	193
LUTYPE6	192
MAGTAPE	20
RESSYS	208
SDLC	176
SEQDISK	18
SYSTEM3	161
SYSTEM7	2
SYS370	164
SYS7BSCA	166
TCONSOLE	8
TELETYPE	34
TTCAM	80
TWX3335	33
T1050	36
T1053	74
T2260L	65
T2260R	72
T2265	76
T2740	40
T2741BCD	43
T2741COR	42
T2770	130
T2780	132
T2980	134
T3275R	146
T3277L	153
T3277R	145
T3284L	155
T3284R	147
T3286L	156
T3286R	148
T3600BI	138
T3601	177
T3614	178
T3650ATT	186
T3650PIPE	184
T3650USER	187
T3653HOST	185
T3735	136
T3740	137
T3780	133
T3790	180
T3790SCSP	182
T3790UP	181
T7770	1
VIDEOTERM	64

Numeric sequence

1	T7770
2	SYSTEM7
8	TCONSOLE
18	SEQDISK
20	MAGTAPE
24	CDRDLPRT
32	HARDCOPY
34	TELETYPE
33	TWX3335
36	T1050
40	T2740
42	T2741COR
43	T2741BCD
64	VIDEOTERM
65	T2260L
72	T2260R
74	T1053
76	T2265
80	TTCAM
128	BISYNCH
130	T2770
132	T2780
133	T3780
134	T2980
136	T3735
137	T3740
138	T3600BI
145	T3277R
146	T3275R
147	T3284R
148	T3286R
153	T3277L
155	T3284L
156	T3286L
160	BIPROG
161	SYSTEM3
164	SYS370
166	SYS7BSCA
176	SDLC
177	T3601
178	T3614
180	T3790
181	T3790UP
182	T3790SCSP
184	T3650PIPE
185	T3653HOST
186	T3650ATT
187	T3650USER
189	CONTNLU
190	INTACTLU
191	BATCHLU
192	LUTYPE6
193	LUTYPE4
208	RESSYS
209	ISCMCONV
210	LUCMODGRP
211	LUCSESS

Appendix B. EXEC interface block (EIB) response and function codes

Response codes of EXEC CICS commands

After the execution of an EXEC CICS command, fields EIBRESP and EIBRCODE are set to indicate whether the command executed successfully, or whether a CICS condition was raised.

Each possible value of EIBRESP relates directly to a specific condition, no matter which command caused the condition to be raised. This is not true for EIBRCODE values: both the value and the byte of EIBRCODE in which it is set depend on which command was issued.

The following sections list the conditions that are applicable to the EXEC CICS commands described in this book, their corresponding RESP values (decimal), the associated EIBRCODE values (hexadecimal), and the transaction abend codes (if any).

EXEC CICS DISCARD, INQUIRE, PERFORM, and SET commands

The first word of EIBRCODE for these commands is always set equal to the hexadecimal equivalent of the RESP value; the remaining bytes are set to X'00'.

Condition	RESP Value	EIBRCODE (Byte 3)	Abend code
DSNNOTFOUND	93	5D	AEX1
DUPREC	14	0E	AEIN
END	83	53	AEXK
FILENOTFOUND	12	0C	AEIL
ILLOGIC	21	15	AEIU
INVREQ	16	10	AEIP
IOERR	17	11	AEIQ
JIDERR	43	2B	AEYG
LENGERR	22	16	AEIV
MODELIDERR	95	5F	AEX3
NOSPACE	18	12	AEIR
NOSTG	42	2A	-
NOTAUTH	70	46	AEY7
NOTFND	13	0D	AEIM
PARTNERIDERR	97	61	AEX5
PGMIDERR	27	1B	AEI0
PROFILEIDERR	98	62	AEX6
QIDERR	44	2C	AEYH
SYSBUSY	59	3B	-
SYSIDERR	53	35	AEYQ
TASKIDERR	91	5B	AEXX
TCIDERR	92	5C	AEX0
TERMIDERR	11	0B	AEIK
TRANSIDERR	28	1C	AEI1
USERIDERR	69	45	AEYX
VOLIDERR	71	47	AEXV

EXEC CICS DISABLE, ENABLE, and EXTRACT EXIT commands

Conditions that can be raised by the ENABLE, DISABLE, and EXTRACT EXIT commands are INVEXITREQ and NOTAUTH. There are no conditions associated with the RESYNC command.

Condition	RESP Value	EIBRCODE (Byte 3)	Abend code
INVEXITREQ	63	80	AEY0
NOTAUTH	70	46	AEY7

Function codes of EXEC CICS commands

The function code (field EIBFN) is a hexadecimal value that identifies the command most recently issued by a task. The format of the EIBFN field is as follows:

```
ASM      CL2
COBOL   PIC X(2)
PL/I    CHAR (2)
C       CHAR variable name(2);
```

The function codes of the commands described in this book are listed below. For information about other function codes, see the *CICS/ESA Problem Determination Guide*.

Command	Code
ACQUIRE TERMINAL	86 02
COLLECT STATISTICS	70 08
+ CREATE CONNECTION	30 0E
+ CREATE FILE	30 14
+ CREATE LSRPOOL	30 16
+ CREATE MAPSET	30 04
+ CREATE PARTITIONSET	30 06
+ CREATE PARTNER	30 18
+ CREATE PROFILE	30 0A
+ CREATE PROGRAM	30 02
+ CREATE SESSIONS	30 12
+ CREATE TERMINAL	30 10
+ CREATE TRANCLASS	30 1A
+ CREATE TRANSACTION	30 08
+ CREATE TYPETERM	30 0C
DISCARD AUTINSTMODEL	42 10
DISCARD FILE	4C 10
DISCARD PARTNER	44 10
DISCARD PROFILE	46 10
DISCARD PROGRAM	4E 10
DISCARD TRANCLASS	5E 18
DISCARD TRANSACTION	50 10
INQUIRE AUTINSTMODEL	42 02
INQUIRE AUTOINSTALL	68 12
INQUIRE CONNECTION	58 02
INQUIRE DSNAME	7A 02
INQUIRE DUMPDS	66 02
INQUIRE EXITPROGRAM	88 02
INQUIRE FILE	4C 02
INQUIRE IRC	6E 02

Command	Code	Code	Command
INQUIRE JOURNALNUM	60 02	42 02	INQUIRE AUTINSTMODEL
INQUIRE MODENAME	5A 02	42 10	DISCARD AUTINSTMODEL
INQUIRE MONITOR	70 12	44 02	INQUIRE PARTNER
INQUIRE NETNAME	52 16	44 10	DISCARD PARTNER
INQUIRE NETNAME	52 06	46 02	INQUIRE PROFILE
INQUIRE PARTNER	44 02	46 10	DISCARD PROFILE
INQUIRE PROFILE	46 02	4C 02	INQUIRE FILE
INQUIRE PROGRAM	4E 02	4C 04	SET FILE
INQUIRE REQID	8A 02	4C 10	DISCARD FILE
INQUIRE STATISTICS	70 02	4E 02	INQUIRE PROGRAM
INQUIRE STORAGE	5E 08	4E 04	SET PROGRAM
INQUIRE SYSDUMPCODE	66 22	4E 10	DISCARD PROGRAM
INQUIRE SYSTEM	54 02	50 02	INQUIRE TRANSACTION
INQUIRE TASK	5E 02	50 04	SET TRANSACTION
INQUIRE TCLASS	5E 12	50 10	DISCARD TRANSACTION
INQUIRE TDQUEUE	5C 02	52 02	INQUIRE TERMINAL
INQUIRE TERMINAL	52 12	52 04	SET TERMINAL
INQUIRE TERMINAL	52 02	52 06	INQUIRE NETNAME
INQUIRE TRACEDEST	78 02	52 08	SET NETNAME
INQUIRE TRACEFLAG	78 12	52 12	INQUIRE TERMINAL
INQUIRE TRACETYPE	78 22	52 14	SET TERMINAL
INQUIRE TRANCLASS	5E 1A	52 16	INQUIRE NETNAME
INQUIRE TRANDUMPCODE	66 12	54 02	INQUIRE SYSTEM
INQUIRE TRANSACTION	50 02	54 04	SET SYSTEM
INQUIRE TSQUEUE	80 02	58 02	INQUIRE CONNECTION
INQUIRE VOLUME	62 02	58 04	SET CONNECTION
INQUIRE VTAM	68 02	5A 02	INQUIRE MODENAME
PERFORM DUMP	7E 04	5A 04	SET MODENAME
PERFORM RESETTIME	72 02	5C 02	INQUIRE TDQUEUE
PERFORM SECURITY	64 02	5C 04	SET TDQUEUE
PERFORM SHUTDOWN	76 02	5E 02	INQUIRE TASK
PERFORM STATISTICS	70 06	5E 04	SET TASK
SET AUTOINSTALL	68 14	5E 08	INQUIRE STORAGE
SET CONNECTION	58 04	5E 1A	INQUIRE TRANCLASS
SET DSNAME	7A 04	5E 1C	SET TRANCLASS
SET DUMPDS	66 04	5E 12	INQUIRE TCLASS
SET FILE	4C 04	5E 14	SET TCLASS
SET IRC	6E 04	5E 18	DISCARD TRANCLASS
SET JOURNALNUM	60 04	60 02	INQUIRE JOURNALNUM
SET MODENAME	5A 04	60 04	SET JOURNALNUM
SET MONITOR	70 14	62 02	INQUIRE VOLUME
SET NETNAME	52 08	62 04	SET VOLUME
SET PROGRAM	4E 04	64 02	PERFORM SECURITY
SET STATISTICS	70 04	66 02	INQUIRE DUMPDS
SET SYSDUMPCODE	66 24	66 04	SET DUMPDS
SET SYSTEM	54 04	66 12	INQUIRE TRANDUMPCODE
SET TASK	5E 04	66 14	SET TRANDUMPCODE
SET TCLASS	5E 14	66 22	INQUIRE SYSDUMPCODE
SET TDQUEUE	5C 04	66 24	SET SYSDUMPCODE
SET TERMINAL	52 14	68 02	INQUIRE VTAM
SET TERMINAL	52 04	68 04	SET VTAM
SET TRACEDEST	78 04	68 12	INQUIRE AUTOINSTALL
SET TRACEFLAG	78 14	68 14	SET AUTOINSTALL
SET TRACETYPE	78 24	6E 02	INQUIRE IRC
SET TRANCLASS	5E 1C	6E 04	SET IRC
SET TRANDUMPCODE	66 14	70 02	INQUIRE STATISTICS
SET TRANSACTION	50 04	70 04	SET STATISTICS
SET VOLUME	62 04	70 06	PERFORM STATISTICS
SET VTAM	68 04	70 08	COLLECT STATISTICS
		70 12	INQUIRE MONITOR

Code	Command		
70 14	SET MONITOR	78 24	SET TRACETYPE
72 02	PERFORM RESETTIME	7A 02	INQUIRE DSNAME
76 02	PERFORM SHUTDOWN	7A 04	SET DSNAME
78 02	INQUIRE TRACEDEST	7E 04	PERFORM DUMP
78 04	SET TRACEDEST	80 02	INQUIRE TSQUEUE
78 12	INQUIRE TRACEFLAG	86 02	ACQUIRE TERMINAL
78 14	SET TRACEFLAG	88 02	INQUIRE EXITPROGRAM
78 22	INQUIRE TRACETYPE	8A 02	INQUIRE REQID

Appendix C. EXEC CICS CREATE RESP2 values

Most of the RESP2 values issued by the EXEC CICS CREATE command are associated with a message that is written to transient data queue CSMT. The RESP2 values and the corresponding message numbers are shown in Table 4 below. For this command, the fullword EIBRESP2 field is regarded as a structure containing two halfwords. The low-order halfword always contains an error number. The high-order halfword sometimes contains another number to help you to identify the error. Sometimes this number is the offset *n* in the ATTRIBUTES string at which the error was detected. Sometimes it is the keyword number *k* for which the error was detected. For a list of the keyword numbers, see pages Table 5 on page 221, Table 6 on page 222, and Table 7 on page 223.

Table 4 (Page 1 of 4). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
Codes caused by syntactical errors		
<i>n</i> ,400	DFHCA5211	A misplaced delimiter occurs in ATTRIBUTES. The invalid delimiter is at offset <i>n</i> in the ATTRIBUTES string.
<i>n</i> ,401	DFHCA5204	A keyword specified within ATTRIBUTES is invalid. The invalid keyword is at offset <i>n</i> in the ATTRIBUTES string.
<i>n</i> ,402	DFHCA5212, DFHCA5213	A keyword within ATTRIBUTES cannot be uniquely identified from its abbreviation. The invalid keyword is at offset <i>n</i> in the ATTRIBUTES string.
<i>k</i> ,403	DFHCA5501	A required keyword is omitted. The omitted keyword has code <i>k</i> in Table 5, Table 6, and Table 7.
404	DFHCA5529	A required keyword is omitted. The omitted keyword must be selected from two mutually exclusive keywords, as specified in the associated message.
<i>k</i> ,405	DFHCA5504	One specified keyword requires another one to be specified. The omitted keyword has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,406	DFHCA5206	A keyword occurs more than once within ATTRIBUTES. The duplicate keyword has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,407	DFHCA5503, DFHCA5506	Conflicting keywords are specified. The keyword causing the conflict has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,410	DFHCA5210, DFHCA5519, DFHCA5521, DFHCA5522, DFHCA5526, DFHCA5528, DFHCA5532	An invalid operand is supplied for a keyword within ATTRIBUTES. The keyword in error has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,411	DFHCA5207	An operand is supplied for a keyword that does not need one. The keyword in error has code <i>k</i> in Table 5, Table 6, and Table 7.

Table 4 (Page 1 of 4). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
<i>k</i> ,412	DFHCA5205	A required operand for a keyword within ATTRIBUTES is omitted. The keyword in error has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,413	DFHCA5517	The operands of two or more keywords conflict with one another. The first conflicting keyword detected has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,414	DFHCA5507	The value of the operand of a keyword within ATTRIBUTES is too small. The keyword in error has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,415	DFHCA5513	In the pair of values specified as the operand of a keyword within ATTRIBUTES, the second value must not exceed the first. The keyword in error has code <i>k</i> in Table 5, Table 6, and Table 7.
<i>k</i> ,416	DFHCA5509	An invalid operand is supplied for a keyword within ATTRIBUTES. The value of the operand must be different from the name of the resource. The keyword in error has code <i>k</i> in Table 5, Table 6, and Table 7.
417	DFHCA5523, DFHCA5535	The specified resource cannot be created with this command. The resource name is reserved for CICS use.
418	DFHCA5527	CICS internal programs (whose names begin with DFH) cannot be given attributes that specify remote execution.
<i>k</i> ,419	DFHCA5217	A closing parenthesis has been omitted from a DESCRIPTION keyword within ATTRIBUTES. The keyword in error (DESCRIPTION) has code <i>k</i> in Table 5, Table 6, and Table 7.
Codes caused by errors deleting existing resources		
500	DFHAM4803, DFHAM4842, DFHAM4834, DFHAM4896, DFHZC5913	Install failed because the resource is currently in use.
501	DFHAM4841	Install failed because definition of <i>restype rename</i> is in use by task no. <i>taskno</i> (transaction id. <i>tranid</i>).
501	DFHZC5980	Resource <i>resource</i> is in use by task <i>taskid</i> Transaction <i>tranid</i>
502	DFHZC6304	Deletion of remote terminal <i>termid</i> failed because it is in use by another transaction.
503	DFHZC5915	Deletion of <i>restype id</i> failed. It needs to be set out of service.
504	DFHAM4899, DFHZC5998	Install specified a resource that cannot be replaced.
505	DFHZC5916	Deletion of terminal <i>termid</i> failed. It has pending DFHZCP activity.

Table 4 (Page 2 of 4). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
505	DFHZC5918	Deletion of terminal <i>termid</i> Console <i>consname</i> failed. It has pending DFHZCP activity.
506	DFHZC5914	Deletion of terminal <i>termid</i> found another deletion of it in progress.
506	DFHZC5937	Deletion of modename <i>modename</i> found another deletion of it in progress.
507	DFHZC5902	Deletion of terminal <i>termid</i> failed. BMS Paging session still active.
508	DFHZC5917	Deletion of terminal <i>termid</i> failed. Error message writer still active.
509	DFHZC5904	Deletion of terminal <i>termid</i> failed. CEDF is still active.
510	DFHZC5941	Install for terminal <i>termid</i> failed. Console <i>consname</i> has a conversation outstanding.
511	DFHZC5907	Deletion of remote shipped terminal failed for connection <i>cccc</i> .
512	DFHZC5925	Deletion of connection <i>cccc</i> failed. Its AID-Chains are not empty
513	DFHZC5929	Deletion of connection <i>cccc</i> failed. It is in use by <i>n</i> indirect connections.
514	DFHZC5938	Deletion of modename <i>modename</i> failed. Unable to delete session(s).
515	DFHZC5951	Deletion of connection <i>ssss</i> failed. Unable to delete sessions.
516	DFHZC5945	Deletion of sessions <i>ssss</i> failed. Connection <i>cccc</i> is defined to IRC.
517	DFHZC5952	Deletion of terminal <i>termid</i> failed. It needs to be SET RELEASED.
518	DFHZC5969	Deletion of dependent modename(s) failed for connection <i>modename</i> .
519	DFHZC5974	Deletion of pool <i>pppp</i> failed. Unable to delete pool entries.
520	DFHZC5979	Deletion of pool <i>pppp</i> failed. It still has session <i>termid</i> .
520	DFHZC5982	Deletion of pool <i>pppp</i> failed. Pool entry is in use for <i>termid</i> .
521	DFHZC5958	Install failed for <i>xxxx</i> . This is the name of the local system, which must not be replaced.
522	DFHZC5940	Install for terminal <i>termid</i> failed. Error console cannot be deleted.
523	DFHZC5989	Deletion of resource <i>resource</i> failed. Remote deletion in connection <i>cccc</i> failed.
524	DFHZC5943	MRO connection <i>conname</i> cannot be deleted because IRC is open.

Codes caused by errors in installing the new resource

600	DFHTO6000	The definition for TERMINAL <i>termdef</i> refers to an undefined TYPETERM <i>termtype</i> .
600	DFHTO6001	The definition for pooled TERMINAL <i>termdef</i> refers to an undefined TYPETERM <i>termtype</i> .
601	DFHTO6002	The definition for SESSIONS <i>sesdef</i> refers to an undefined CONNECTION <i>condef</i> .
601	DFHZC5911	Install for resource <i>resource</i> failed. Connection <i>cccc</i> not found.
601	DFHZC5932	Install for modename <i>modename</i> failed. Connection <i>cccc</i> not found.

Table 4 (Page 2 of 4). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
602	DFHZC5962	Install for resource <i>resource</i> failed. Modename parameter not found.
603	DFHZC5906	Install failed because <i>xxxx</i> is not a permitted value for a terminal or connection name.
604	DFHZC5933	Install for modename <i>modename</i> failed. Connection <i>cccc</i> is not valid here.
607	DFHAM4870	Install failed for program <i>progrname</i> - language RPG is not supported under MVS.
620	DFHZC5912	Install for terminal <i>termid</i> failed. It is incompatible with connection <i>cccc</i> .
620	DFHZC5949	Install for sessions <i>ssss</i> failed. It is incompatible with connection <i>cccc</i> .
621	DFHZC5900	System <i>sysid</i> has shipped definitions but connection <i>cccc</i> is not known to this system.
622	DFHZC5921	Install of terminal <i>termid</i> failed. VTAM support not loaded.
622	DFHZC5988	Install for resource <i>resource</i> failed. VTAM support not generated.
623	DFHZC5909	Install of resource <i>resource</i> failed. Call to DFHIRP <i>irp_function</i> <i>Return_code</i> did not succeed, See DFHIRSDS for return code.
624	DFHZC5931	Install for modename <i>modename</i> failed. Maximum number of APPC sessions would have been exceeded.
625	DFHZC5973	Install for sessions <i>ssss</i> failed. Max session-count reached for modename <i>modename</i> .
626	DFHZC5955	SESNUMB greater than DLTHRED in the SIT (<i>nnnn</i>).
627	DFHZC5934	Install for modename <i>modename</i> failed. Single-session connection <i>cccc</i> is already in use.
628	DFHZC5936	Install for modename <i>modename</i> failed. Connection <i>cccc</i> has active modegroup <i>xxxx</i> .
629	DFHZC5939	Install for <i>name</i> failed. Duplicate session- or modegroup-name for connection <i>sysid</i> .
630	DFHZC5946	Install for sessions <i>ssss</i> failed. Connection <i>cccc</i> is defined to IRC.
631	DFHZC5948	Install for sessions <i>ssss</i> failed. Connection <i>cccc</i> is not suitable for IRC.
632	DFHZC5954	Install for resource <i>resource</i> failed. Unable to install sessions component.
633	DFHZC5963	<i>operation</i> RUSIZE <i>xxxx</i> from terminal <i>termid</i> was greater than TYPETERM RUSIZE <i>yyyy</i> .
634	DFHZC5967	Install for modename <i>modename</i> failed. Unable to install sessions.
635	DFHZC5968	Unable to install LU Services Manager for modename <i>modename</i> .
636	DFHZC5981	Pool <i>pppp</i> not found.
637	DFHZC5985	Install for resource <i>resource</i> failed. Unable to install connection component.
638	DFHTO6003	TERMINAL <i>termdef</i> specifies CONSOLE but refers to TYPETERM <i>termtype</i> which does not specify DEVICE=CONSOLE.

Table 4 (Page 3 of 4). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
639	DFHTO6004	TERMINAL <i>termdef</i> does not specify CONSOLE but refers to TYPETERM <i>termtype</i> which specifies DEVICE=CONSOLE.
640	DFHTO6005	PRINTER or ALTPRINTER for TERMINAL <i>termdef</i> is invalid for the DEVICE specified in TYPETERM <i>termtype</i> .
641	DFHTO6006	PRINTERCOPY or ALTPRINTERCOPY for TERMINAL <i>termdef</i> is invalid for the DEVICE specified in TYPETERM <i>termtype</i> .
642	DFHTO6007	AUTINSTMODEL YES ONLY for TERMINAL <i>termdef</i> is invalid for the DEVICE specified in TYPETERM <i>termtype</i> .
643	DFHTO6008	
644	DFHTO6009	The definition for SESSIONs <i>sesdef</i> refers to CONNECTION <i>condef</i> which specifies a different PROTOCOL.
645	DFHTO6010	The definition for SESSIONs <i>sesdef</i> must specify PROTOCOL LU61 as it refers to an MRO CONNECTION <i>condef</i> .
646	DFHTO6011	SESSIONs <i>sesdef</i> must specify both SENDCOUNT and RECEIVECOUNT as it refers to an MRO CONNECTION <i>condef</i> .
647	DFHTO6013	No SESSIONs definition refers to CONNECTION <i>condef</i> .
648	DFHTO6014	POOL is required for TERMINAL <i>termdef</i> as it refers to TYPETERM <i>typedef</i> which specifies SESSIONTYPE=PIPELINE.
649	DFHTO6015	TRANSACTION for TERMINAL <i>termdef</i> is invalid for the DEVICE specified in TYPETERM <i>typedef</i> .
650	DFHTO6016	The MRO CONNECTION <i>condef</i> is referenced by more than one SESSIONs definition, including <i>sesdef</i> .
651	DFHTO6017	REMOTESYSTEM for TERMINAL <i>termid</i> is invalid for the DEVICE specified in TYPETERM <i>typeterm</i> .
652	DFHTO6018	TERMINAL <i>termid</i> refers to TYPETERM <i>typeterm</i> which has an invalid ALTSCREEN.
653	DFHTO6020	SESSIONs <i>sesdef</i> refers to single-session CONNECTION <i>condef</i> but has an invalid MAXIMUM option specified.
654	DFHTO6023	Connection definition @BCH detected. Batch shared database connections are not supported.
655	DFHTO6025	The definition for LU6.1 SESSIONs <i>sesdef</i> specifies a send or receive count with no prefix.
656	DFHZA6301	Install for <i>tttt</i> failed. Duplicate netname <i>netname</i> for resource <i>rrrr</i> found.
657	DFHZA6302	Install for connection <i>cccc</i> failed. Duplicate netname <i>netname</i> for resource <i>rrrr</i> found.
658	DFHZA6303	Install for <i>tttt</i> failed. Duplicate netname <i>netname</i> found.

Table 4 (Page 3 of 4). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
660	DFHZA6331	Install for connection <i>tttt</i> failed. Non-VTAM terminal with same name already exists.
660	DFHZA6332	Install for terminal <i>tttt</i> failed. Non-VTAM terminal with same name already exists.
661	DFHZA5950	Install for terminal <i>termid</i> failed. Console <i>consname</i> already exists.
662	DFHZA6310	Install for terminal <i>termid</i> failed. Console <i>consname</i> must be defined by ID not name.
663	DFHZA6311	Install for terminal <i>termid</i> failed. Console ID <i>conslid</i> does not map to a console name known to MVS.
664	DFHZA6330	Install for <i>tttt</i> failed. LDCLIST parameter <i>ldclist</i> not found.
665	DFHZA6333	INSTALL for modename <i>modename</i> failed. Zero sessions specified.
666	DFHAM4833	Resource cannot be installed with specified userid because of a security error.
667	DFHZA6362	Install for terminal <i>portname</i> with userid <i>userid</i> failed because the preset userid has been revoked.
668	DFHZA6363	Install for terminal <i>portname</i> with userid <i>userid</i> failed because the preset userid's group access has been revoked.
669	DFHZA6364	Install for terminal <i>portname</i> with userid <i>userid</i> failed because the ESM returned an unrecognized response.
670	DFHZA6365	Install for terminal <i>portname</i> with userid <i>userid</i> failed because the external security manager is inactive.
671	DFHZA6366	Install for terminal <i>portname</i> with userid <i>userid</i> failed because the userid is not authorized to access this CICS system.
672	DFHZA6367	Install for terminal <i>termid</i> with userid <i>userid</i> failed because the SECLABEL check failed.
673	DFHZA6368	Install for terminal <i>portname</i> with userid <i>userid</i> failed because the external security manager is quiesced.
674	DFHZA6369	Install for terminal <i>portname</i> failed because national language <i>langcode</i> is invalid.
675	DFHZA6370	Install for terminal <i>portname</i> failed because national language <i>langcode</i> is unavailable.
676	DFHZA6371	Install for terminal <i>portname</i> with userid <i>userid</i> failed because the userid is not authorized to use this portname.
677	DFHZA5944	Install for <i>type(id)</i> has failed. It would make a loop of connection definitions.
Codes caused by CICS internal logic errors		
900	DFHTO6012	The catalog dataset is not available. RDO function is restricted.
901	DFHAM4872	Unable to connect to CICS catalog.
902	DFHAM4873	Unable to disconnect the CICS catalog.
903	DFHZA6209	Invalid ZC catalog request code <i>xxxx</i> .
904	DFHZA6212	Level mismatch with catalog record. DFHBS <i>xxx</i> .

Table 4 (Page 4 of 4). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
905	DFHAM4898	Install failed because sufficient storage
	DFHXC5901	could not be obtained.
906	DFHXC6200	Could not obtain DWE storage.
907	DFHXC6203	Unable to obtain DWE action-list
		storage.
908	DFHXC6214	Unable to obtain recovery record
		storage.
950	DFHXC6202	Pattern <i>pattern</i> not valid for builder.
951	DFHXC6204	Illegal subpattern definition <i>pattern</i> .
952	DFHXC6205	Illegal subpattern definition <i>pattern</i> .
953	DFHXC6206	Pattern <i>pattern</i> not valid for destroy.
954	DFHXC6207	Catalog key too long or zero. Pattern
		<i>pattern</i> .
955	DFHXC6213	Recovery record abandoned. Key is
		<i>key</i> .
956	DFHXC6341	Loop or ABEND has been detected in
		<i>inmodule</i> by module <i>bymodule</i> .

Table 5 (Page 1 of 2). Keywords associated with keyword numbers. CREATE CONNECTION through CREATE MAPSET.

Keyword number	CONNECTION	FILE	EXEC CICS CREATE command names	MAPSET
1	CONNECTION	FILE	LSRPOOL	MAPSET
5	NETNAME	(RESSECNUM) #	LSRPOOL	(RSL) #
6	INDSYS	DSNAME	MAXKEYLENGTH	DESCRIPTION
7	SECURITYNAME	RECORDSIZE	SHARELIMIT	
8	(BINDPASSWORD) #	KEYLENGTH	STRINGS	
9		JOURNAL	DATA512	
10	REMOTESYSTEM	REMOTESYSTEM	DATA1K	
11	REMOTENAME	REMOTENAME	DATA2K	
12	DESCRIPTION	PASSWORD	DATA4K	
13	QUEUELIMIT	LSRPOOLID	DATA8K	
14	MAXQTIME	STRINGS	DATA12K	
15		DATABUFFERS	DATA16K	
16		INDEXBUFFERS	DATA20K	
17		FWDRECOVLOG	DATA24K	
18		DESCRIPTION	DATA28K	
19		NSRGROUP	DATA32K	
20		MAXNUMRECS	LSRPOOLID	
21			DESCRIPTION	
22			INDEX512	
23			INDEX1K	
24			INDEX2K	
25			INDEX4K	
26			INDEX8K	
27	REMOTESYSNET		INDEX12K	
28			INDEX16K	
29			INDEX20K	
30			INDEX24K	
31			INDEX28K	
32			INDEX32K	
33			HSDATA4K	
34			HSDATA8K	
35			HSDATA12K	
36			HSDATA16K	
37			HSDATA20K	
38			HSDATA24K	
39			HSDATA28K	
40			HSDATA32K	
41			HSINDEX4K	
42			HSINDEX8K	
43			HSINDEX12K	
44			HSINDEX16K	
45			HSINDEX20K	
46			HSINDEX24K	
47			HSINDEX28K	
48			HSINDEX32K	
97	INSERVICE	STATUS		STATUS
98	AUTOCONNECT	RECOVERY		
99	PROTOCOL	OPENTIME		
100	ACCESSMETHOD	DISPOSITION		RESIDENT
101	SINGLESESS	ADD		USAGE
102	DATASTREAM	BROWSE		USELPACOPY
103	RECORDFORMAT	DELETE		
104	ATTACHSEC	READ		
105	BINDSECURITY	UPDATE		
106	CONNTYPE	JNLSYNCREAD		
107	PSRECOVERY	JNLSYNWRITE		
108		JNLREAD		
109		JNLUPDATE		
110	USEDFTUSER	JNLADD		
111		DSNSHARING		

Keywords in parentheses, such as (RSL), are obsolete but are retained for cross-release compatibility.

Table 5 (Page 2 of 2). Keywords associated with keyword numbers. CREATE CONNECTION through CREATE MAPSET.

Keyword number	EXEC CICS CREATE command names			
	CONNECTION	FILE	LSRPOOL	MAPSET
112		RECORDFORMAT		
113		TABLE		
114		BACKUPTYPE		
115		RLSACCESS		
116		READINTEG		

Note:

1 Keyword number 1 always refers to the first operand of the CREATE command; that is, the resource being created.

Table 6. Keywords associated with keyword numbers. CREATE PARTITIONSET through CREATE SESSIONS.

Keyword number	EXEC CICS CREATE command names				
	PARTITIONSET	PARTNER	PROFILE	PROGRAM	SESSIONS
1 1	PARTITIONSET	PARTNER	PROFILE	PROGRAM	SESSIONS
5	(RSL) #	NETNAME	MODENAME	(RSL) #	CONNECTION
6	DESCRIPTION	DESCRIPTION	JOURNAL	DESCRIPTION	SESSNAME
7		NETWORK	NEPCCLASS	REMOTESYSTEM	NETNAMEQ
8		PROFILE	RTIMOUT	REMOTENAME	MODENAME
9		TPNAME	DESCRIPTION	TRANSID	MAXIMUM
10		XTPNAME			
11					RECEIVEPFX
12					RECEIVECOUNT
13					SENDPFX
14					SENDCOUNT
15					(OPERID) #
16					(OPERPRIORITY) #
17					(OPERRSL) #
18					(OPERSECURITY) #
19					USERID
20					SENDSIZE
21					RECEIVESIZE
22					(TRANSACTION) #
23					SESSPRIORITY
24					USERAREALEN
25					IOAREALEN
27					NEPCCLASS
28					DESCRIPTION
97	STATUS			STATUS	(INSERVICE) #
98			SCRNSIZE	LANGUAGE	AUTOCONNECT
99			MSGJRNL	RELOAD	BUILDCHAIN
100	RESIDENT		MSGINTEG	RESIDENT	PROTOCOL
101	USAGE		ONEWTE	USAGE	RELREQ
102	USELPACOPY		PROTECT	USELPACOPY	DISCREQ
103			DVSUPRT	CEDF	RECOVOPTION
104			INBFMH	DATALOCATION	(RECOVNOTIFY) #
105			RAQ	EXECKEY	
106			LOGREC		
107			PRINTERCOMP	EXECUTIONSET	
108			CHAINCONTROL		
109			UCTRAN		

Note:

1 Keyword number 1 always refers to the first operand of the CREATE command; that is, the resource being created.

Keywords in parentheses, such as (RSL), are obsolete but are retained for cross-release compatibility.

Table 7 (Page 1 of 2). Keywords associated with keyword numbers. CREATE TERMINAL through CREATE TYPETERM.

Keyword number

EXEC CICS CREATE command names

	TERMINAL	TRANCLASS	TRANSACTION	TYPETERM
1 1	TERMINAL	TRANCLASS	TRANSACTION	TYPETERM
5		MAXACTIVE	(RSL) #	DEVICE
6	AUTINSTNAME	DESCRIPTION	PROGRAM	TERMMODEL
7	TYPETERM	PURGETHRESH	TWASIZE	SESSIONTYPE
8	NETNAME		PROFILE	
9	CONSOLE		PARTITIONSET	LDCLIST
10	REMOTESYSTEM		REMOTESYSTEM	DEFSCREEN
11	REMOTENAME		REMOTENAME	
12	MODENAME		PRIORITY	ALTSCREEN
13	PRINTER		(TCLASS) #	
14	ALTPRINTER		TASKREQ	CGCSGID
15	(OPERID) #		XTRANID	
16	(OPERPRIORITY) #		DTIMOUT	SENDSIZE
17	(OPERRSL) #		(TRANSEC) #	RECEIVESIZE
18	(OPERSECURITY) #		TRPROF	LOGMODE
19	USERID		(PRIMEDSIZE) #	PAGESIZE
20	POOL		ALIAS	
21	TASKLIMIT		DESCRIPTION	ALTPAGE
22	TRANSACTION		TPNAME	
23	TERMPRIORITY		XTPNAME	ALTSUFFIX
24			TRANCLASS	USERAREALEN
25			RUNAWAY	IOAREALEN
26	SECURITYNAME		WAITTIME	
27	BINDPASSWORD			NEPCCLASS
28	DESCRIPTION			DESCRIPTION
29	NATLANG			
30	CONSNAME			
33	REMOTESYSNET			
97	INSERVICE		STATUS	
98	PRINTERCOPY		LOCALQ	AUTOCONNECT
99	ALTPRINTCOPY		INDOUBT	SHIPPABLE
100	AUTINSTMODEL		RESTART	APLKYBD
101			SPURGE	APLTEXT
102	ATTACHSEC		TPURGE	AUDIBLEALARM
103	BINDSECURITY		DUMP	COLOR
104	USEDFLTUSER		EXTSEC	COPY
105			RESSEC	DUALCASEKYBD
106			TRACE	EXTENDEDDS
107			DYNAMIC	HILIGHT
108			CMDSEC	KATAKANA
109			TASKDATALOC	LIGHTPEN
110			TASKDATAKEY	MSRCONTROL
111			STORAGECLEAR	OBFORMAT
112			SHUTDOWN	PARTITIONS
113			ISOLATE	PRINTADAPTER
114			CONFDATA	PROGSYMBOLS
115				VALIDATION
116				FORMFEED
117				HORIZFORM
118				VERTICALFORM
119				TEXTKYBD
120				TEXTPRINT
121				QUERY
122				OUTLINE
123				SOSI
124				BACKTRANS
125				ASCII
126				BRACKET
127				FMHPARM
128				OBOPERID

Keywords in parentheses, such as (RSL), are obsolete but are retained for cross-release compatibility.

Table 7 (Page 2 of 2). Keywords associated with keyword numbers. CREATE TERMINAL through CREATE TYPETERM.

Keyword number	EXEC	CICS	CREATE	command names
	TERMINAL	TRANCLASS	TRANSACTION	TYPETERM
129				AUTOPAGE
130				ERRLASTLINE
131				ERRINTENSIFY
132				ERRCOLOR
133				ERRHILIGHT
134				ATI
135				CREATESESS
136				RELREQ
137				DISCREQ
138				SIGNOFF
139				ROUTEDMSGS
140				LOGONMSG
141				BUILDCHAIN
142				UCTRAN
143				TTI
144				RECOVPTION
145				RECOVNOTIFY
146				XRFSIGNOFF

Note:

- 1 Keyword number 1 always refers to the first operand of the CREATE command; that is, the resource being created.

Index

A

- absolute expression 10
- access to system information
 - INQUIRE STORAGE command 96
- ACCESSMETHOD option
 - INQUIRE CONNECTION command 64
 - INQUIRE DSNNAME command 69
 - INQUIRE FILE command 75
 - INQUIRE TERMINAL command 115
- ACQSTATUS option
 - INQUIRE CONNECTION command 64
 - INQUIRE TERMINAL command 115
 - SET CONNECTION command 150
 - SET MODENAME command 166
 - SET TERMINAL command 184
- ACQUIRE TERMINAL command 21
 - conditions 22
- ACTION option
 - SET DSNNAME command 156
 - SET SYSDUMPCODE command 174
 - SET TRANDUMPCODE command 195
 - SET VOLUME command 199
- ACTIVE option
 - INQUIRE MODENAME command 82
 - INQUIRE TRANCLASS command 127
- ADD option
 - INQUIRE FILE command 75
 - SET FILE command 160
- ADDRESS option
 - INQUIRE STORAGE command 96
- AFTER option
 - INQUIRE REQID command 92
- AKP option
 - INQUIRE SYSTEM command 100
 - SET SYSTEM command 176
- ALIGNED attribute
 - PL/I 9
- ALL option
 - PERFORM STATISTICS command 145
- ALTPAGEHT option
 - INQUIRE TERMINAL command 115
- ALTPAGEWD option
 - INQUIRE TERMINAL command 115
- ALTPRINTER option
 - INQUIRE TERMINAL command 116
 - SET TERMINAL command 184
- ALTPRTCOPIST option
 - INQUIRE TERMINAL command 116
 - SET TERMINAL command 184
- ALTSCRNHT option
 - INQUIRE TERMINAL command 116
- ALTSCRNWD option
 - INQUIRE TERMINAL command 116
- ALTSUFFIX option
 - INQUIRE TERMINAL command 116
- APLKYBDST option
 - INQUIRE TERMINAL command 116
- APLTEXTST option
 - INQUIRE TERMINAL command 116
- ARCHSTATUS option
 - INQUIRE JOURNALNUM command 79
- argument lengths 10
- argument values
 - assembler language 10
 - C/370 9
 - COBOL 9
 - PL/I 9
- ASCII option
 - INQUIRE TERMINAL command 116
- assembler language
 - argument values 10
- AT option
 - INQUIRE REQID command 92
- ATIFACILITY option
 - INQUIRE TDQUEUE command 111
 - SET TDQUEUE command 181
- ATISTATUS option
 - INQUIRE TERMINAL command 116
 - SET TERMINAL command 184
- ATITERMID option
 - INQUIRE TDQUEUE command 111
 - SET TDQUEUE command 181
- ATITRANID option
 - INQUIRE TDQUEUE command 111
 - SET TDQUEUE command 181
- ATIUSERID option
 - INQUIRE TDQUEUE command 111
 - SET TDQUEUE command 181
- ATTRIBUTES option
 - CREATE CONNECTION command 28
 - CREATE FILE command 30
 - CREATE LSRPOOL command 32
 - CREATE MAPSET command 33
 - CREATE PARTITIONSET command 35
 - CREATE PARTNER command 37
 - CREATE PROFILE command 39
 - CREATE PROGRAM command 41
 - CREATE SESSIONS command 43
 - CREATE TERMINAL command 46
 - CREATE TRANCLASS command 47
 - CREATE TRANSACTION command 50
 - CREATE TYPETERM command 52

ATTRLEN option
 CREATE CONNECTION command 28
 CREATE FILE command 30
 CREATE LSRPOOL command 32
 CREATE MAPSET command 33
 CREATE PARTITIONSET command 35
 CREATE PARTNER command 37
 CREATE PROFILE command 39
 CREATE PROGRAM command 41
 CREATE SESSIONS command 44
 CREATE TERMINAL command 46
 CREATE TRANCLASS command 47
 CREATE TRANSACTION command 50
 CREATE TYPETERM command 52

AUDALARMST option
 INQUIRE TERMINAL command 116

authorization failures 11

AUTINSTMODEL option
 DISCARD AUTINSTMODEL command 55
 INQUIRE AUTINSTMODEL command 62

AUTINSTMODEL, DISCARD command 55
 AUTINSTMODEL, INQUIRE command 62

AUTOCONNECT option
 INQUIRE CONNECTION command 64
 INQUIRE MODENAME command 82
 INQUIRE TERMINAL command 116

AUTOINSTALL option
 COLLECT STATISTICS command 24
 PERFORM STATISTICS command 145

AUTOINSTALL, INQUIRE command 63
 AUTOINSTALL, SET command 148

automatic installation of terminals 63

AUXSTATUS option
 INQUIRE TRACEDEST command 124
 SET TRACEDEST command 189

AVAILABLE option
 INQUIRE MODENAME command 82
 SET MODENAME command 166

AVAILSTATUS option
 INQUIRE VOLUME command 136
 SET VOLUME command 199

B

BACKTRANSST option
 INQUIRE TERMINAL command 116

BACKUPTYPE option
 INQUIRE DSNAME command 69

BASEDSNAME option
 INQUIRE DSNAME command 69
 INQUIRE FILE command 75

batch backout utility 69

BKOUTSTATUS option
 INQUIRE DSNAME command 69
 SET DSNAME command 157

BLOCKFORMAT option
 INQUIRE FILE command 75
 INQUIRE TDQUEUE command 112

BLOCKKEYLEN option
 INQUIRE FILE command 75

BLOCKSIZE option
 INQUIRE FILE command 75

BROWSE option
 INQUIRE FILE command 75
 SET FILE command 160

browsing
 AUTINSTMODEL entries 62
 CONNECTION entries 64
 FILE entries 75
 MODENAME entries 81
 PARTNER entries 85
 PROFILE entries 86
 PROGRAM entries 88
 TDQUEUE entries 111
 TERMINAL entries 115
 TRANCLASS entries 127
 TRANDUMPCODE entries 129
 TRANSACTION entries 131
 VOLUME entries 136

browsing resource definitions 13

browsing rules 15

BUSY option
 SET FILE command 160

C

C/370 language
 argument values 9

CALLER option
 PERFORM DUMP command 140

CALLERLENGTH option
 PERFORM DUMP command 140

CANCEL option
 SET CONNECTION command 152

CDSASIZE option
 INQUIRE SYSTEM command 100

CECI transaction 3

CEDF transaction 3

CEDFSTATUS option
 INQUIRE PROGRAM command 88
 SET PROGRAM command 169

CEMT transaction
 function provided by INQUIRE and SET commands 3

CETR transaction 3
 function provided by INQUIRE and SET commands 3

char-expr argument, CICS command format 5

CICS-supplied security 3

CICS-value data area (CVDA) 7

CICSSTATUS option
 INQUIRE SYSTEM command 100

- CICSSYS option
 - INQUIRE SYSTEM command 100
- CICSVR MVS/ESA 69
- CMDPROTECT option
 - INQUIRE SYSTEM command 100
- CMDSEC option
 - INQUIRE TASK command 104
 - INQUIRE TRANSACTION command 131
- COBOL
 - argument values 9
- COBOLTYPE option
 - INQUIRE PROGRAM command 88
- COLLECT STATISTICS
 - conditions 25
- COLLECT STATISTICS command 23
- COLORST option
 - INQUIRE TERMINAL command 117
- command interpreter transaction (CECI) 3
- command security checking 11
- command, CREATE FILE 29
- commands
 - format, arguments 4
- COMPID option
 - INQUIRE TRACETYPE command 126
 - SET TRACETYPE command 193
- COMPLETE option
 - CREATE CONNECTION command 28
 - CREATE TERMINAL command 46
- conditions
 - ACQUIRE TERMINAL command 22
 - COLLECT STATISTICS command 25
 - CREATE CONNECTION command 28
 - CREATE FILE command 30
 - CREATE LSRPOOL command 32
 - CREATE MAPSET command 33
 - CREATE PARTITIONSET command 35
 - CREATE PARTNER command 37
 - CREATE PROFILE command 40
 - CREATE PROGRAM command 42
 - CREATE SESSIONS command 44
 - CREATE TERMINAL command 46
 - CREATE TRANCLASS command 47
 - CREATE TRANSACTION command 50
 - CREATE TYPETERM command 52
 - DISABLE PROGRAM command 54
 - DISCARD AUTINSTMODEL command 55
 - DISCARD FILE command 55
 - DISCARD PARTNER command 56
 - DISCARD PROFILE command 56
 - DISCARD PROGRAM command 57
 - DISCARD TRANCLASS command 57
 - DISCARD TRANSACTION command 58
 - ENABLE PROGRAM command 60
 - EXTRACT EXIT command 61
 - INQUIRE AUTINSTMODEL command 62
 - INQUIRE AUTOINSTALL command 63

- conditions (*continued*)
 - INQUIRE command 15
 - INQUIRE CONNECTION command 67
 - INQUIRE DELETSHIPED 68
 - INQUIRE DSNNAME command 70
 - INQUIRE DUMPDS command 71
 - INQUIRE EXITPROGRAM command 73
 - INQUIRE FILE command 78
 - INQUIRE IRC command 79
 - INQUIRE JOURNALNUM command 80
 - INQUIRE MODENAME command 82
 - INQUIRE MONITOR command 84
 - INQUIRE PARTNER command 86
 - INQUIRE PROFILE command 86
 - INQUIRE PROGRAM command 90
 - INQUIRE REQID command 93
 - INQUIRE STATISTICS command 95
 - INQUIRE STORAGE command 97
 - INQUIRE SYSDUMPCODE command 98
 - INQUIRE SYSTEM command 103
 - INQUIRE TASK command 108
 - INQUIRE TASK LIST command 109
 - INQUIRE TCLASS command 110
 - INQUIRE TDQUEUE command 113
 - INQUIRE TERMINAL command 123
 - INQUIRE TRACEDEST command 125
 - INQUIRE TRACEFLAG command 126
 - INQUIRE TRACETYPE command 127
 - INQUIRE TRANCLASS command 128
 - INQUIRE TRANDUMPCODE command 129
 - INQUIRE TRANSACTION command 133
 - INQUIRE TSQUEUE 135
 - INQUIRE VOLUME command 137
 - INQUIRE VTAM command 139
 - PERFORM DUMP command 140
 - PERFORM RESETTIME command 141
 - PERFORM SECURITY REBUILD command 142
 - PERFORM SHUTDOWN command 143
 - PERFORM STATISTICS RECORD command 146
 - RESYNC ENTRYNAME command 148
 - SET AUTOINSTALL command 148
 - SET CONNECTION command 153
 - SET DELETSHIPED command 155
 - SET DSNNAME command 157
 - SET DUMPDS command 159
 - SET FILE command 162
 - SET IRC command 164
 - SET JOURNALNUM command 165
 - SET MODENAME command 166
 - SET MONITOR command 168
 - SET NETNAME command 169
 - SET PROGRAM command 170
 - SET STATISTICS command 173
 - SET SYSDUMPCODE command 175
 - SET SYSTEM command 178
 - SET TASK command 179

conditions (*continued*)

- SET TCLASS command 180
- SET TDQUEUE command 181
- SET TERMINAL command 188
- SET TRACEDEST command 190
- SET TRACEFLAG command 192
- SET TRACETYPE command 193
- SET TRANCLASS command 194
- SET TRANDUMPCODE command 196
- SET TRANSACTION command 198
- SET VOLUME 199
- SET VTAM command 201

CONNECTION

- SET CONNECTION command 150

CONNECTION option

- COLLECT STATISTICS command 24
- CREATE CONNECTION command 28
- INQUIRE CONNECTION command 65
- INQUIRE MODENAME command 82
- PERFORM STATISTICS command 145
- SET MODENAME command 166

CONNECTION, CREATE command 27

CONNECTION, INQUIRE command 64

CONNECTION, SET command 150

CONNECTST option

- INQUIRE EXITPROGRAM command 72

CONNSTATUS option

- INQUIRE CONNECTION command 65
- SET CONNECTION command 150

CONNTYPE option

- INQUIRE CONNECTION command 65

CONVERSEST option

- INQUIRE MONITOR command 83
- SET MONITOR command 167

COPY option

- INQUIRE PROGRAM command 88
- SET PROGRAM command 170

COPYST option

- INQUIRE TERMINAL command 117

CREATE CONNECTION command 27

- conditions 28

CREATE FILE command 29

- conditions 30

CREATE LSRPOOL command 31

- conditions 32

CREATE MAPSET command 33

- conditions 33

CREATE PARTITIONSET command 35

- conditions 35

CREATE PARTNER command 37

- conditions 37

CREATE PROFILE command 39

- conditions 40

CREATE PROGRAM command 41

- conditions 42

- CREATE SESSIONS command 43
 - conditions 44
- CREATE TERMINAL command 45
 - conditions 46
- CREATE TRANCLASS command 47
 - conditions 47
- CREATE TRANSACTION command 49
 - conditions 50
- CREATE TYPETERM command 51
 - conditions 52
- CREATESESS option
 - INQUIRE TERMINAL command 117
 - SET TERMINAL command 184
- creating resource definitions 16
- CURAUXTDS option
 - INQUIRE TRACEDEST command 124
- CURRENT option
 - INQUIRE SYSDUMPCODE command 97
 - INQUIRE TCLASS command 110
 - INQUIRE TRANDUMPCODE command 129
- CURRENTDDS option
 - INQUIRE DUMPDS command 71
- CURRENTVOL option
 - INQUIRE JOURNALNUM command 80
- CURREQS option
 - INQUIRE AUTOINSTALL command 63
- CVDA (CICS-value data area)
 - argument values 5
 - command format 5
 - example code 7
 - listed in numerical and alphabetical order 203
 - on INQUIRE commands 7

D

DAEOPTION option

- INQUIRE SYSDUMPCODE command 98, 174

data set name blocks

data table options

- MAXNUMRECS option on SET FILE command 162
- TABLE option on SET FILE command 162

data types 8

data-area argument

- CICS command format 5

data-areas 5

data-value argument

- CICS command format 5

data-values 5

DATALOCATION option

- INQUIRE PROGRAM command 88

DATASTREAM option

- INQUIRE TERMINAL command 117

DATE option

- INQUIRE VOLUME command 136

defining exits 18

DEFPAGEHT option
 INQUIRE TERMINAL command 117
 DEFPAGEWD option
 INQUIRE TERMINAL command 117
 DEFSCRNHT option
 INQUIRE TERMINAL command 117
 DEFSCRNWD option
 INQUIRE TERMINAL command 117
 DELETE option
 INQUIRE FILE command 75
 SET FILE command 161
 DELETSHIPED, INQUIRE command 68
 DELETSHIPED, PERFORM command 139
 DELETSHIPED, SET command 154
 DEVICE option
 INQUIRE TERMINAL command 117
 DFLTUSER option
 INQUIRE SYSTEM command 100
 DISABLE PROGRAM command 53
 conditions 54
 examples for global user exits 54
 DISABLED CVDA value
 INQUIRE AUTOINSTALL command 63
 DISCARD AUTINSTMODEL command 55
 conditions 55
 DISCARD FILE command 55
 conditions 55
 DISCARD option
 CREATE CONNECTION command 28
 CREATE TERMINAL command 46
 DISCARD PARTNER command 56
 conditions 56
 DISCARD PROFILE command 56
 conditions 56
 DISCARD PROGRAM command 57
 conditions 57
 DISCARD TRANCLASS command 57
 conditions 57
 DISCARD TRANSACTION command 58
 conditions 58
 discarding resources
 resource definitions 17, 55
 DISCREQST option
 INQUIRE TERMINAL 184
 INQUIRE TERMINAL command 117
 DISKASTATUS option
 INQUIRE JOURNALNUM command 80
 DISKBSTATUS option
 INQUIRE JOURNALNUM command 80
 DISKXSTATUS option
 INQUIRE JOURNALNUM command 80
 DISPATCHABLE option
 INQUIRE TASK LIST command 109
 DISPATCHER option
 COLLECT STATISTICS command 24
 PERFORM STATISTICS command 145
 DISPOSITION option
 INQUIRE FILE command 75
 SET FILE command 161
 DSALIMIT option
 INQUIRE SYSTEM command 100
 SET SYSTEM command 176
 DSNAME option
 INQUIRE DSNAME command 70
 INQUIRE FILE command 75
 SET DSNAME command 157
 SET FILE command 161
 DSNAME, INQUIRE command 69
 DSNAME, SET command 156
 DTB option
 COLLECT STATISTICS command 24
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131
 PERFORM STATISTICS command 145
 DTIMEOUT option
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131
 DTRPROGRAM option
 INQUIRE SYSTEM command 101
 SET SYSTEM command 176
 DUALCASEST option
 INQUIRE TERMINAL command 117
 dump data sets 71
 DUMP option
 PERFORM SHUTDOWN command 143
 DUMP, PERFORM command 140
 DUMPCODE option
 PERFORM DUMP command 140
 DUMPDS, INQUIRE command 71
 DUMPDS, SET command 158
 DUMPID option
 PERFORM DUMP command 140
 DUMPING option
 INQUIRE SYSTEM command 101
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131
 SET SYSTEM command 177
 SET TRANSACTION command 197
 DUMPSCOPE option
 INQUIRE SYSDUMPCODE command 98, 174
 INQUIRE TRANDUMPCODE command 129
 SET TRANDUMPCODE command 195

E

ECDSASIZE option
 INQUIRE SYSTEM command 101
 EDSALIMIT option
 INQUIRE SYSTEM command 101
 SET SYSTEM command 177
 EDSASIZE option
 INQUIRE SYSTEM command 101

ELEMENT option
 INQUIRE STORAGE command 96
 ELEMENTLIST option
 INQUIRE STORAGE command 96
 EMPTYSTATUS option
 INQUIRE FILE command 76
 INQUIRE TDQUEUE command 112
 INQUIRE VOLUME command 136
 SET FILE command 161
 ENABLE PROGRAM command 58
 conditions 60
 examples for global user exits 60
 examples for task-related user exits 61
 ENABLED CVDA value
 INQUIRE AUTOINSTALL command 63
 ENABLESTATUS option
 INQUIRE AUTOINSTALL command 63
 INQUIRE FILE command 76
 INQUIRE TDQUEUE command 112
 SET FILE command 161
 SET TDQUEUE command 181
 ENDOFDAY option
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 ENDOFDAYHRS option
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 ENDOFDAYMINS option
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 ENDOFDAYSECS option
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 ENTRY option
 ENABLE PROGRAM command 58
 INQUIRE EXITPROGRAM command 72
 ENTRYNAME option
 DISABLE PROGRAM command 53
 ENABLE PROGRAM command 59
 EXTRACT EXIT command 61
 INQUIRE EXITPROGRAM command 72
 RESYNC command 147
 ENTRYNAME, RESYNC command 147
 ENTRYPOINT option
 INQUIRE PROGRAM command 88
 ERDSASIZE option
 INQUIRE SYSTEM command 101
 ESM (external security manager) 3
 ESMRESP option
 PERFORM SECURITY REBUILD command 142
 EUDSASIZE option
 INQUIRE SYSTEM command 101
 EVENTCLASS option
 INQUIRE MONITOR command 83
 SET MONITOR command 167
 EXCEPTCLASS option
 INQUIRE MONITOR command 83
 SET MONITOR command 167
 EXCLUSIVE option
 INQUIRE FILE command 76
 SET FILE command 161
 EXEC CICS command format 4
 EXEC CICS CREATE
 RESP2 values 217
 EXEC CICS INQUIRE command
 See INQUIRE commands
 EXEC CICS PERFORM command
 See PERFORM commands
 EXEC CICS SET command
 See SET commands
 EXECKEY option
 INQUIRE PROGRAM command 88
 execution diagnostic facility transaction (CEDF) 3
 EXECUTIONSET option
 INQUIRE PROGRAM command 88
 SET PROGRAM command 170
 exit names 18
 EXIT option
 DISABLE PROGRAM command 53
 ENABLE PROGRAM command 59
 INQUIRE EXITPROGRAM command 72
 exit-related commands 18
 EXIT, EXTRACT command 61
 EXITALL option
 DISABLE PROGRAM command 53
 EXITPROGRAM option
 INQUIRE EXITPROGRAM command 73
 EXITPROGRAM, INQUIRE command 72
 exits
 defining 18
 EXITTRACING option
 INQUIRE CONNECTION command 66
 INQUIRE TERMINAL command 117
 SET CONNECTION command 151
 SET NETNAME command 169
 SET TERMINAL command 184
 EXTENDEDSSST option
 INQUIRE TERMINAL command 118
 external security manager (ESM) 3
 EXTRACT EXIT command 61
 conditions 61

F
 FACILITY option
 INQUIRE TASK command 105
 FACILITYTYPE option
 INQUIRE TASK command 105
 FEPI option
 PERFORM STATISTICS command 145

FILE option
 COLLECT STATISTICS command 24
 CREATE FILE command 30
 DISCARD FILE command 55
 INQUIRE FILE command 76
 PERFORM STATISTICS command 145
 SET FILE command 161
 FILE, DISCARD command 55
 FILE, INQUIRE command 74
 FILE, SET command 160
 FILECOUNT option
 INQUIRE DSNAME command 70
 filename argument, CICS command format 5
 FLENGTH option
 INQUIRE STORAGE command 96
 INQUIRE TSQUEUE command 134
 FMHPARMST option
 INQUIRE TERMINAL command 118
 FMHSTATUS option
 INQUIRE REQID command 92
 FORCECANCEL option
 SET CONNECTION command 152
 format rules 5
 FORMATEDF option
 DISABLE PROGRAM command 53
 ENABLE PROGRAM command 59
 FORMATEDFST option
 INQUIRE EXITPROGRAM command 73
 FORMFEEDST option
 INQUIRE TERMINAL command 118
 FREQUENCY option
 INQUIRE MONITOR command 83
 SET MONITOR command 167
 FREQUENCYHRS option
 INQUIRE MONITOR command 83
 SET MONITOR command 168
 FREQUENCYMIN option
 INQUIRE MONITOR command 83
 SET MONITOR command 168
 FREQUENCYSEC option
 INQUIRE MONITOR command 84
 SET MONITOR command 168
 function shipping, not available for SP commands 3
 FWDRECOVLOG option
 INQUIRE DSNAME command 70
 FWDRECSTATUS option
 INQUIRE FILE command 76

G

GAENTRYNAME option
 ENABLE PROGRAM command 59
 INQUIRE EXITPROGRAM command 73
 GALENGTH option
 ENABLE PROGRAM command 59
 EXTRACT EXIT command 61

GALENGTH option (*continued*)
 INQUIRE EXITPROGRAM command 73
 GASET option
 EXTRACT EXIT command 61
 GAUSECOUNT option
 INQUIRE EXITPROGRAM command 73
 GCHARS option
 INQUIRE TERMINAL command 118
 GCODES option
 INQUIRE TERMINAL command 118
 GMMLENGTH option
 INQUIRE SYSTEM command 101
 SET SYSTEM command 177
 GMMTEXT option
 INQUIRE SYSTEM command 101
 SET SYSTEM command 177
 GMMTRANID option
 INQUIRE SYSTEM command 101
 GRNAME option
 INQUIRE VTAM command 138
 GRSTATUS option
 INQUIRE VTAM command 138
 GTFSTATUS option
 INQUIRE TRACEDEST command 124
 SET TRACEDEST command 190

H

HFORMST option
 INQUIRE TERMINAL command 118
 HILIGHTST option
 INQUIRE TERMINAL command 118
 HOLDSTATUS option
 INQUIRE PROGRAM command 89
 HOURS option
 INQUIRE REQID command 92

I

IDLIST option
 RESYNC command 147
 IDLISTLENGTH option
 RESYNC command 147
 IGNORE (null values) 10
 IMMEDIATE option
 PERFORM SHUTDOWN command 143
 INDIRECTNAME option
 INQUIRE TDQUEUE command 112
 INITIALDDS option
 INQUIRE DUMPDS command 71
 SET DUMPDS command 158
 INITSTATUS option
 INQUIRE SYSTEM command 101
 INQUIRE and SET commands
 examples
 Assembler 8
 C 8

INQUIRE and SET commands (*continued*)
 examples (*continued*)
 COBOL 8
 PL/I 8
 null values 10
 INQUIRE AUTINSTMODEL command 62
 conditions 62
 INQUIRE AUTOINSTALL command 63
 conditions 63
 INQUIRE command, browse
 conditions 15
 INQUIRE commands
 AUTINSTMODEL 62
 AUTOINSTALL 63
 CONNECTION 64
 DELETSHIPED 68
 DSNAME 69
 DUMPDS 71
 EXITPROGRAM 72
 FILE 74
 IRC 79
 JOURNALNUM 79
 MODENAME 81
 MONITOR 83
 NETNAME 85
 PARTNER 85
 PROFILE 86
 PROGRAM 87
 REQID 91
 STATISTICS 94
 STORAGE 96
 SYSDUMPCODE 97
 SYSTEM 99
 TASK 104
 TASK LIST 109
 TCLASS 110
 TDQUEUE 111
 TERMINAL 114
 TRACEDEST 124
 TRACEFLAG 125
 TRACETYPE 126
 TRANCLASS 127
 TRANDUMPCODE 128
 TRANSACTION 130
 TSQUEUE 134
 VOLUME 136
 VTAM 138
 INQUIRE CONNECTION command 64
 conditions 67
 INQUIRE DELETSHIPED command 68
 conditions 68
 INQUIRE DSNAME command 69
 conditions 70
 INQUIRE DUMPDS command 71
 INQUIRE EXITPROGRAM command 72
 conditions 73

INQUIRE FILE command 74
 INQUIRE IRC command 79
 conditions 79
 INQUIRE JOURNALNUM command 79
 conditions 80
 INQUIRE MODENAME command 81
 conditions 82
 INQUIRE MONITOR command 83
 conditions 84
 INQUIRE NETNAME command 85
 INQUIRE PARTNER command 85
 conditions 86
 INQUIRE PROFILE command 86
 conditions 86
 PROFILE 86
 INQUIRE PROGRAM command 87
 conditions 90
 INQUIRE REQID command 91
 conditions 93
 INQUIRE STATISTICS command 94
 conditions 95
 INQUIRE STORAGE command 96
 conditions 97
 INQUIRE SYSDUMPCODE command 97
 conditions 98
 INQUIRE SYSTEM command 99
 conditions 103
 INQUIRE TASK command 104
 conditions 108
 INQUIRE TASK LIST command 109
 conditions 109
 INQUIRE TCLASS command 110
 conditions 110
 INQUIRE TDQUEUE command 111
 conditions 113
 INQUIRE TERMINAL command 114
 conditions 123
 INQUIRE TRACEDEST command 124
 conditions 125
 INQUIRE TRACEFLAG command 125
 conditions 126
 INQUIRE TRACETYPE command 126
 conditions 127
 INQUIRE TRANCLASS command 127
 conditions 128
 INQUIRE TRANDUMPCODE command 128
 conditions 129
 INQUIRE TRANSACTION command 130
 conditions 133
 INQUIRE TSQUEUE command 134
 conditions 135
 INQUIRE VOLUME command 136
 conditions 137
 INQUIRE VTAM command 138
 conditions 139

inquiry commands 13
 integer-expr argument, CICS command format 5
 INTERVAL option
 INQUIRE REQID command 92
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 INTERVALHRS option
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 INTERVALLMINS option
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 INTERVALSECS option
 INQUIRE STATISTICS command 94
 SET STATISTICS command 173
 INTSTATUS option
 INQUIRE TRACEDEST command 124
 SET TRACEDEST command 190
 IOTYPE option
 INQUIRE TDQUEUE command 112
 IRC, INQUIRE command 79
 IRC, SET command 163
 IRCBATCH option
 COLLECT STATISTICS command 24
 PERFORM STATISTICS command 145
 ISOLATEST option
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131

J

JOBNAME option
 INQUIRE SYSTEM command 101
 JOURNALNUM option
 COLLECT STATISTICS command 24
 INQUIRE FILE command 76
 INQUIRE JOURNALNUM command 80
 INQUIRE VOLUME command 136
 PERFORM STATISTICS command 145
 SET JOURNALNUM command 164
 SET VOLUME command 199
 JOURNALNUM, INQUIRE command 79
 JOURNALNUM, SET command 164
 JTYPE option
 INQUIRE JOURNALNUM command 80

K

KATAKANAST option
 INQUIRE TERMINAL command 118
 KEYLENGTH option
 INQUIRE FILE command 76
 KEYPOSITION option
 INQUIRE FILE command 76

L

label argument, CICS command format 5
 LANGDEDUCED option
 INQUIRE PROGRAM command 89
 LANGUAGE option
 INQUIRE PROGRAM command 89
 LASTRESET option
 COLLECT STATISTICS command 24
 LASTRESETHRS option
 COLLECT STATISTICS command 24
 LASTRESETMIN option
 COLLECT STATISTICS command 24
 LASTRESETSEC option
 COLLECT STATISTICS command 24
 LENGTH option
 default (PL/I) 8
 INQUIRE PROGRAM command 89
 INQUIRE REQID command 92
 LENGTHLIST option
 INQUIRE STORAGE command 96
 LIGHTPENST option
 INQUIRE TERMINAL command 118
 LINKEDITMODE option
 ENABLE PROGRAM command 59
 LISTSIZE option
 INQUIRE TASK LIST command 109
 literal constants 10
 LOADPOINT option
 INQUIRE PROGRAM command 89
 LOCATION option
 INQUIRE TSQUEUE command 135
 LONGDATE option
 INQUIRE VOLUME command 136
 LPASTATUS option
 INQUIRE PROGRAM command 89
 LSRPOOL option
 COLLECT STATISTICS command 25
 CREATE LSRPOOL command 32
 PERFORM STATISTICS command 145
 LSRPOOL, CREATE command 31
 LSRPOOLID option
 INQUIRE FILE command 76
 SET FILE command 161

M

MAPNAME option
 INQUIRE TERMINAL command 118
 SET TERMINAL command 184
 MAPSET option
 CREATE MAPSET command 33
 MAPSET, CREATE command 33
 MAPSETNAME option
 INQUIRE TERMINAL command 119
 SET TERMINAL command 184

MAXACTIVE option
 INQUIRE TRANCLASS command 127
 SET TRANCLASS command 194
 MAXIMUM option
 INQUIRE MODENAME command 82
 INQUIRE SYSDUMPCODE command 98
 INQUIRE TCLASS command 110
 INQUIRE TRANDUMPCODE command 129
 SET SYSDUMPCODE command 175
 SET TCLASS command 180
 SET TRANDUMPCODE command 195
 MAXITEMLEN option
 INQUIRE TSQUEUE command 135
 MAXNUMRECS option
 INQUIRE FILE command 76
 SET FILE command 162
 MAXREQS option
 INQUIRE AUTOINSTALL command 63
 SET AUTOINSTALL command 148
 MAXTASKS option
 INQUIRE SYSTEM command 101
 SET SYSTEM command 177
 MAXWINNERS option
 INQUIRE MODENAME command 82
 MINITEMLEN option
 INQUIRE TSQUEUE command 135
 MINUTES option
 INQUIRE REQID command 92
 MODENAME option
 INQUIRE MODENAME command 82
 INQUIRE TERMINAL command 119
 SET MODENAME command 166
 MODENAME, INQUIRE command 81
 MODENAME, SET command 165
 MONITOR option
 COLLECT STATISTICS command 25
 PERFORM STATISTICS command 145
 MONITOR, INQUIRE command 83
 MONITOR, SET command 166
 MROBATCH option
 INQUIRE SYSTEM command 101
 SET SYSTEM command 177
 MSRCONTROLST option
 INQUIRE TERMINAL command 119

N

name argument, CICS command format 5
 NATLANG option
 INQUIRE TERMINAL command 119
 NATURE option
 INQUIRE TERMINAL command 119
 NETNAME option
 INQUIRE CONNECTION command 66
 INQUIRE PARTNER command 85
 INQUIRE TERMINAL command 119

NETNAME option (*continued*)
 SET NETNAME command 169
 NETNAME, INQUIRE command 85
 NETNAME, SET command 169
 NETWORK option
 INQUIRE PARTNER command 85
 NEWMAXTASKS option
 SET SYSTEM command 177
 NEXTTIME option
 INQUIRE STATISTICS command 94
 NEXTTIMEHRS option
 INQUIRE STATISTICS command 94
 NEXTTIMEMINS option
 INQUIRE STATISTICS command 95
 NEXTTIMESECS option
 INQUIRE STATISTICS command 95
 NEXTTRANSID option
 INQUIRE TERMINAL command 119
 SET TERMINAL command 185
 NODE TARGET option
 COLLECT STATISTICS command 25
 NOHANDLE
 option 4, 10
 NOQUEUE option
 ACQUIRE TERMINAL command 21
 NORESTART option
 PERFORM SHUTDOWN command 143
 NOTAPPLIC 10
 notation, syntax 4
 NOTAUTH condition 11
 null values 10
 NUMELEMENTS option
 INQUIRE STORAGE command 96
 NUMEXITS option
 INQUIRE EXITPROGRAM command 73
 NUMITEMS option
 INQUIRE TDQUEUE command 112
 INQUIRE TSQUEUE command 134

O

OBFORMATST option
 INQUIRE TERMINAL 185
 INQUIRE TERMINAL command 119
 OBJECT option
 INQUIRE DSNAME command 70
 INQUIRE FILE command 76
 OBOPERIDST option
 INQUIRE TERMINAL command 120
 OLDESTPART option
 INQUIRE JOURNALNUM command 80
 OPENSTATUS option
 INQUIRE DUMPDS command 71
 INQUIRE FILE command 77
 INQUIRE IRC command 79
 INQUIRE JOURNALNUM command 80

OPENSTATUS option (*continued*)
 INQUIRE TDQUEUE command 112
 INQUIRE VOLUME command 136
 INQUIRE VTAM command 138
 SET DUMPDS command 159
 SET FILE command 162
 SET IRC command 163
 SET JOURNALNUM command 164
 SET TDQUEUE command 181
 SET VTAM command 200

OPERID option
 INQUIRE TERMINAL command 120
 SET TERMINAL command 185

OPREL option
 INQUIRE SYSTEM command 101

OPSYS option
 INQUIRE SYSTEM command 101

OUTLINEST option
 INQUIRE TERMINAL command 120

P

PAGEHT option
 INQUIRE TERMINAL command 120
 PAGESSTATUS option
 INQUIRE TERMINAL command 120
 SET TERMINAL command 185
 PAGEWD option
 INQUIRE TERMINAL command 120
 PART option
 INQUIRE VOLUME command 137

PARTITIONSET option
 CREATE PARTITIONSET command 35
 PARTITIONSET, CREATE command 35

PARTITIONSST option
 INQUIRE TERMINAL command 120

PARTNER option
 CREATE PARTNER command 37
 DISCARD PARTNER command 56
 INQUIRE PARTNER command 85

PARTNER, CREATE command 37
 PARTNER, DISCARD command 56
 PARTNER, INQUIRE command 85

PENDSTATUS option
 INQUIRE CONNECTION command 66
 SET CONNECTION command 151

PERFCLASS option
 INQUIRE MONITOR command 84
 SET MONITOR command 168

PERFORM commands
 DELETSHIPED 139
 DUMP 140
 RESETTIME 141
 SECURITY REBUILD 142
 SHUTDOWN 143
 STATISTICS RECORD 144

PERFORM DELETSHIPED command 139
 PERFORM DUMP command 140
 conditions 140
 PERFORM RESETTIME command 141
 conditions 141
 PERFORM SECURITY REBUILD command 142
 conditions 142
 PERFORM SHUTDOWN command 143
 conditions 143
 PERFORM STATISTICS RECORD command 144
 conditions 146

PL/I language
 argument values 9
 LENGTH option default 8

PLT option
 PERFORM SHUTDOWN command 143

pointer arguments 7
 pointer-ref argument, CICS command format 5
 pointer-value argument, CICS command format 5
 POOL option

COLLECT STATISTICS command 25
 POOL TARGET option

COLLECT STATISTICS command 25
 PRINTADAPTST option

INQUIRE TERMINAL command 120
 PRINTCONTROL option

INQUIRE TDQUEUE command 112
 PRINTER option

INQUIRE TERMINAL command 120
 SET TERMINAL command 185

PRIORITY option
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131
 SET TASK command 179
 SET TRANSACTION command 197

PROFILE option
 CREATE PROFILE command 39
 DISCARD PROFILE command 56
 INQUIRE PARTNER command 85
 INQUIRE PROFILE command 86
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131

PROFILE, DISCARD command 56
 PROFILE, INQUIRE command 86

PROG AUTO option
 COLLECT STATISTICS command 25
 PERFORM STATISTICS command 145

PROG AUTOCTLG option
 INQUIRE SYSTEM command 102
 SET SYSTEM command 177

PROG AUTOEXIT option
 INQUIRE SYSTEM command 102
 SET SYSTEM command 177

PROG AUTOINST option
 INQUIRE SYSTEM command 102
 SET SYSTEM command 177

PROGRAM option
 COLLECT STATISTICS command 25
 CREATE PROGRAM command 42
 DISABLE PROGRAM command 53
 DISCARD PROGRAM command 57
 ENABLE PROGRAM command 59
 EXTRACT EXIT command 61
 INQUIRE AUTOINSTALL command 63
 INQUIRE PROGRAM command 89
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131
 PERFORM STATISTICS command 145
 SET AUTOINSTALL command 148
 SET PROGRAM command 170
 PROGRAM, CREATE command 41
 PROGRAM, DISABLE command 53
 PROGRAM, DISCARD command 57
 PROGRAM, ENABLE command 58
 PROGRAM, INQUIRE command 87
 PROGRAM, SET command 169
 PROGSYMBOLST option
 INQUIRE TERMINAL command 120
 PROGTYPE option
 INQUIRE PROGRAM command 89
 PROTOCOL option
 INQUIRE CONNECTION command 66
 PRTCOPYST option
 INQUIRE TERMINAL command 120
 SET TERMINAL command 185
 PRTYAGING option
 INQUIRE SYSTEM command 102
 SET SYSTEM command 177
 PSDINTERVAL option
 INQUIRE VTAM command 138
 SET VTAM 200
 PSDINTHRS option
 INQUIRE VTAM command 139
 SET VTAM 201
 PSDINTMINS option
 INQUIRE VTAM command 139
 SET VTAM 201
 PSDINTSECS option
 INQUIRE VTAM command 139
 SET VTAM 201
 PURGEABILITY option
 INQUIRE TASK command 105
 INQUIRE TRANSACTION command 131
 SET TRANSACTION command 197
 PURGETHRESH option
 INQUIRE TRANCLASS command 127
 SET TRANCLASS command 194
 PURGETYPE option
 SET CONNECTION command 151
 SET TASK command 179
 SET TERMINAL command 185

Q

QALL option
 ACQUIRE TERMINAL command 21
 QNOTETAB option
 ACQUIRE TERMINAL command 21
 QSESSLIM option
 ACQUIRE TERMINAL command 21
 QUALIFIER option
 RESYNC command 147
 QUERY SECURITY command 13
 QUERYST option
 INQUIRE TERMINAL command 120
 QUEUE option
 INQUIRE REQID command 92
 QUEUED option
 INQUIRE TRANCLASS command 127

R

RACF (resource access control facility) 3
 RDSASIZE option
 INQUIRE SYSTEM command 102
 READ option
 INQUIRE FILE command 77
 SET FILE command 162
 RECEIVECOUNT option
 INQUIRE CONNECTION command 66
 RECORDFORMAT option
 INQUIRE FILE command 77
 INQUIRE TDQUEUE command 113
 RECORDING option
 INQUIRE STATISTICS command 95
 SET STATISTICS command 173
 RECORDLENGTH option
 INQUIRE TDQUEUE command 113
 RECORDNOW option
 SET STATISTICS command 173
 RECORDSIZE option
 INQUIRE FILE command 77
 RECOVSTATUS option
 INQUIRE DSNAME command 70
 INQUIRE FILE command 77
 INQUIRE TDQUEUE command 113
 RECUNITID option
 INQUIRE TASK command 106
 REENTPROTECT option
 INQUIRE SYSTEM command 102
 RELEASE option
 INQUIRE SYSTEM command 102
 relocatable expression 10
 RELREQ option
 ACQUIRE TERMINAL command 21
 RELREQST option
 INQUIRE TERMINAL command 121
 SET TERMINAL command 186

- RELTYPE option
 - INQUIRE FILE command 77
- remote definition, not retrievable or updatable 3
- REMOTENAME option
 - INQUIRE FILE command 77
 - INQUIRE PROGRAM command 89
 - INQUIRE TASK command 106
 - INQUIRE TDQUEUE command 113
 - INQUIRE TERMINAL command 121
 - INQUIRE TRANSACTION command 132
- REMOTESYSTEM option
 - INQUIRE FILE command 77
 - INQUIRE PROGRAM command 89
 - INQUIRE TASK command 106
 - INQUIRE TDQUEUE command 113
 - INQUIRE TERMINAL command 121
 - INQUIRE TRANSACTION command 132
- REMOTETABLE option
 - INQUIRE FILE command 77
- removing 156
- REQID option
 - INQUIRE REQID command 92
- REQID, INQUIRE command 91
- REQTYPE option
 - INQUIRE REQID command 92
- RESCOUNT option
 - INQUIRE PROGRAM command 90
- RESETNOW option
 - PERFORM STATISTICS command 145
 - SET STATISTICS command 173
- RESEETIME, PERFORM command 141
- resource access control facility (RACF) 3
- resource security checking 12
- resources
 - class (ESM) 11
- RESP
- RESP and RESP2 options 11
 - values returned 11
- RESP options 11
- RESP2
 - option 10
- RESP2 options 11
- RESP2 values
 - EXEC CICS CREATE 217
- RESSEC option
 - INQUIRE TASK command 106
 - INQUIRE TRANSACTION command 132
- RESYNC command
 - ENTRYNAME 147
 - IDLIST 147
 - IDLISTLENGTH 147
 - QUALIFIER 147
- RESYNC ENTRYNAME command 147
 - conditions 148
- ROUTING option
 - INQUIRE TASK command 106

- ROUTING option (*continued*)
 - INQUIRE TRANSACTION command 132
- RTERMID option
 - INQUIRE REQID command 92
- RTIMEOUT option
 - INQUIRE TASK command 106
 - INQUIRE TRANSACTION command 132
- RTRANSID option
 - INQUIRE REQID command 92
- rules for browsing 15
- RUNAWAY option
 - INQUIRE SYSTEM command 102
 - INQUIRE TASK command 106
 - INQUIRE TRANSACTION command 132
 - SET SYSTEM command 177
 - SET TRANSACTION command 197
- RUNAWAYTYPE option
 - INQUIRE TRANSACTION command 132
 - SET TRANSACTION command 197
- RUNNING option
 - INQUIRE TASK LIST command 109
- RUNSTATUS option
 - INQUIRE TASK command 106

S

- SCANDELAY option
 - INQUIRE SYSTEM command 102
 - SET SYSTEM command 178
- SCREENHEIGHT, see SCRNH
- SCREENWIDTH, see SCRND
- SCRNH option
 - INQUIRE TERMINAL command 121
- SCRNSIZE option
 - INQUIRE TASK command 106
 - INQUIRE TRANSACTION command 132
- SCRND option
 - INQUIRE TERMINAL command 121
- SDSASIZE option
 - INQUIRE SYSTEM command 102
- SECONDS option
 - INQUIRE REQID command 92
- security 11—13
 - command 11
 - NOTAUTH condition 11
 - QUERY SECURITY command 13
 - resource security checking 11
 - security checking by ESM 11
- security check failures 11
- security checking
 - command 11
 - resource 12
 - surrogate 11
 - transaction 11
- SECURITY option
 - INQUIRE TERMINAL command 121

SECURITY REBUILD, PERFORM command 142
 SECURITYMGR option
 INQUIRE SYSTEM command 102
 SENDCOUNT option
 INQUIRE CONNECTION command 66
 SERVSTATUS option
 INQUIRE CONNECTION command 66
 INQUIRE TERMINAL command 121
 SET CONNECTION command 152
 SET TERMINAL command 186
 SESSIONS option
 CREATE SESSIONS command 44
 SESSIONTYPE option
 INQUIRE TERMINAL command 121
 SET AUTOINSTALL command 148
 conditions 148
 SET commands 15
 AUTOINSTALL 148
 CONNECTION 150
 DELETSHIPED 154
 DSNAME 156
 DUMPDS 158
 FILE 160
 IRC 163
 JOURNALNUM 164
 MODENAME 165
 MONITOR 166
 NETNAME 169
 PROGRAM 169
 STATISTICS 172
 SYSDUMPCODE 174
 SYSTEM 176
 TASK 179
 TCLASS 180
 TDQUEUE 180
 TERMINAL 183
 TRACEDEST 189
 TRACEFLAG 191
 TRACETYPE 193
 TRANCLASS 194
 TRANDUMPCODE 195
 TRANSACTION 197
 VOLUME 199
 VTAM 200
 SET CONNECTION command 150
 conditions 153
 SET DELETSHIPED command 154
 conditions 155
 SET DSNAME command 156
 conditions 157
 SET DUMPDS command 158
 conditions 159
 SET FILE command 160
 conditions 162
 SET IRC command 163
 conditions 164
 SET JOURNALNUM command 164
 conditions 165
 SET MODENAME command 165
 conditions 166
 SET MONITOR command 166
 conditions 168
 SET NETNAME command 169
 conditions 169
 SET option
 COLLECT STATISTICS command 25
 INQUIRE REQID command 92
 INQUIRE TASK LIST command 109
 SET PROGRAM command 169
 conditions 170
 SET STATISTICS command 172
 conditions 173
 SET SYSDUMPCODE command 174
 conditions 175
 SET SYSTEM command 176
 conditions 178
 SET TASK command 179
 conditions 179
 SET TCLASS command 180
 conditions 180
 SET TDQUEUE command 180
 conditions 181
 SET TERMINAL command 183
 conditions 188
 SET TRACEDEST command 189
 conditions 190
 SET TRACEFLAG command 191
 conditions 192
 SET TRACETYPE command 193
 conditions 193
 SET TRANCLASS command 194
 conditions 194
 SET TRANDUMPCODE command 195
 conditions 196
 SET TRANSACTION command 197
 conditions 198
 SET VOLUME command 199
 conditions 199
 SET VTAM command 200
 conditions 201
 SETTRANSID option
 INQUIRE TASK LIST command 109
 SHARESTATUS option
 INQUIRE PROGRAM command 90
 SET PROGRAM command 170
 SHUTDOWN option
 DISABLE PROGRAM command 53
 ENABLE PROGRAM command 60
 INQUIRE TRANSACTION command 132
 SET TRANSACTION command 198
 SHUTDOWN, PERFORM command 143

SHUTDOWNST option
 INQUIRE EXITPROGRAM command 73
 SHUTOPTION option
 INQUIRE SYSDUMPCODE command 98
 INQUIRE TRANDUMPCODE command 129
 SET SYSDUMPCODE command 175
 SET TRANDUMPCODE command 195
 SHUTSTATUS option
 INQUIRE SYSTEM command 102
 SIGNONSTATUS option
 INQUIRE TERMINAL command 121
 SINGLESTATUS option
 INQUIRE TRACEFLAG command 125
 SET TRACEFLAG command 191
 SOSIST option
 INQUIRE TERMINAL command 121
 SOSSTATUS option
 INQUIRE SYSTEM command 103
 SPECIAL option
 INQUIRE TRACETYPE command 127
 SET TRACETYPE command 193
 STANDARD option
 INQUIRE TRACETYPE command 127
 SET TRACETYPE command 193
 START option
 ENABLE PROGRAM command 60
 STARTCODE option
 INQUIRE TASK command 106
 starting a browse 14
 STARTSTATUS option
 INQUIRE EXITPROGRAM command 73
 STARTUP option
 INQUIRE SYSTEM command 103
 STARTUPDATE option
 INQUIRE SYSTEM command 103
 STATISTICS RECORD, PERFORM command 144
 STATISTICS, COLLECT command 23
 STATISTICS, INQUIRE command 94
 STATISTICS, SET command 172
 STATS option
 PERFORM STATISTICS command 145
 STATUS option
 INQUIRE MONITOR command 84
 INQUIRE PROGRAM command 90
 INQUIRE TRANSACTION command 132
 SET MONITOR command 168
 SET PROGRAM command 170
 SET TRANSACTION command 198
 STOP option
 DISABLE PROGRAM command 53
 STORAGE option
 COLLECT STATISTICS command 25
 PERFORM STATISTICS command 145
 STORAGE, INQUIRE command 96
 STORAGECLEAR option
 INQUIRE TASK command 107
 STORAGECLEAR option (*continued*)
 INQUIRE TRANSACTION command 132
 STOREPROTECT option
 INQUIRE SYSTEM command 103
 STRINGS option
 INQUIRE FILE command 77
 SET FILE command 162
 SUBSYSTEMID option
 INQUIRE MONITOR command 84
 surrogate security checking 11
 SUSPENDED option
 INQUIRE TASK LIST command 109
 SUSPENDTIME option
 INQUIRE TASK command 107
 SUSPENDTYPE option
 INQUIRE TASK command 107
 SUSPENDVALUE option
 INQUIRE TASK command 107
 SWITCHACTION option
 SET TRACEDEST command 190
 SWITCHSTATUS option
 INQUIRE DUMPDS command 71
 INQUIRE TRACEDEST command 124
 SET DUMPDS command 159
 SET TRACEDEST command 190
 SYNCPOINTST option
 INQUIRE MONITOR command 84
 SET MONITOR command 168
 syntax notation 4
 SYSDUMP option
 PERFORM STATISTICS command 145
 SYSDUMPCODE option
 COLLECT STATISTICS command 25
 SET SYSDUMPCODE command 175
 SYSDUMPCODE, INQUIRE command 97
 SYSDUMPCODE, SET command 174
 SYSDUMPING option
 INQUIRE SYSDUMPCODE command 98
 INQUIRE TRANDUMPCODE command 129
 SET SYSDUMPCODE command 175
 SET TRANDUMPCODE command 196
 SYSEIB
 option 4
 system connections 64
 system programming commands 3
 inquiry 13
 SYSTEM, INQUIRE command 99
 SYSTEM, SET command 176
 systemname argument, CICS command format 5
 SYSTEMSTATUS option
 INQUIRE TRACEFLAG command 125
 SET TRACEFLAG command 192

T

- TABLE option
 - INQUIRE FILE command 78
 - SET FILE command 162
- TABLEMGR option
 - COLLECT STATISTICS command 25
 - PERFORM STATISTICS command 145
- TABLESIZE option
 - INQUIRE TRACEDEST command 125
 - SET TRACEDEST command 190
- TAKEOVER option
 - PERFORM SHUTDOWN command 143
- TALength option
 - ENABLE PROGRAM command 60
 - INQUIRE EXITPROGRAM command 73
- TASK LIST, INQUIRE command 109
- TASK option
 - INQUIRE STORAGE command 96
 - INQUIRE TASK command 107
 - SET TASK command 179
- task-related user exits, restart resynchronization 147
- TASK, INQUIRE command 104
- TASK, SET command 179
- TASKDATAKEY option
 - INQUIRE TASK command 107
 - INQUIRE TRANSACTION command 132
- TASKDATALOC option
 - INQUIRE TASK command 107
 - INQUIRE TRANSACTION command 133
- TASKID option
 - INQUIRE TERMINAL command 121
- TASKSTART option
 - DISABLE PROGRAM command 54
 - ENABLE PROGRAM command 60
- TASKSTARTST option
 - INQUIRE EXITPROGRAM command 73
- TCAMCONTROL option
 - INQUIRE TERMINAL command 122
 - SET TERMINAL command 186
- TCEXITSTATUS option
 - INQUIRE TRACEFLAG command 126
 - SET TRACEFLAG command 192
- TCLASS option
 - COLLECT STATISTICS command 25
 - INQUIRE TASK command 107
 - INQUIRE TRANSACTION command 133
 - PERFORM STATISTICS command 145
 - SET TRANSACTION command 198
- TCLASS, INQUIRE command 110
- TCLASS, SET command 180
- TDQUEUE option
 - COLLECT STATISTICS command 25
 - INQUIRE TDQUEUE command 113
 - PERFORM STATISTICS command 145
 - SET TDQUEUE command 181
- TDQUEUE, INQUIRE command 111
- TDQUEUE, SET command 180
- TERMINAL option
 - ACQUIRE TERMINAL command 21
 - COLLECT STATISTICS command 25
 - CREATE TERMINAL command 46
 - INQUIRE TERMINAL command 122
 - PERFORM STATISTICS command 145
 - SET TERMINAL command 187
- TERMINAL, ACQUIRE command 21
- TERMINAL, CREATE command 46
- TERMINAL, INQUIRE command 114
- TERMINAL, SET command 183
- TERMMODEL option
 - INQUIRE TERMINAL command 122
- TERMPRIORITY option
 - INQUIRE TERMINAL command 122
 - SET TERMINAL command 187
- TERMSTATUS option
 - INQUIRE TERMINAL command 122
 - SET TERMINAL command 187
- TEXTKYBDST option
 - INQUIRE TERMINAL command 122
- TEXTPRINTST option
 - INQUIRE TERMINAL command 122
- TIME option
 - INQUIRE MONITOR command 84
 - INQUIRE REQID command 92
 - INQUIRE SYSTEM command 103
 - INQUIRE VOLUME command 137
 - SET SYSTEM command 178
- timeout delete mechanism 68
- TITLE option
 - PERFORM DUMP command 140
- TITLELENGTH option
 - PERFORM DUMP command 140
- TPNAME option
 - INQUIRE PARTNER command 85
- TPNAMELEN option
 - INQUIRE PARTNER command 86
- TRACEDEST, INQUIRE command 124
- TRACEDEST, SET command 189
- TRACEFLAG, INQUIRE command 125
- TRACEFLAG, SET command 191
- TRACETYPE, INQUIRE command 126
- TRACETYPE, SET command 193
- TRACING option
 - INQUIRE TASK command 108
 - INQUIRE TERMINAL command 122
 - INQUIRE TRANSACTION command 133
 - SET TERMINAL command 187
 - SET TRANSACTION command 198
- TRANCLASS option
 - COLLECT STATISTICS command 25

TRANCLASS option (*continued*)
 CREATE TRANCLASS command 47
 DISCARD TRANCLASS command 57
 INQUIRE TASK command 108
 INQUIRE TRANCLASS command 128
 INQUIRE TRANSACTION command 133
 PERFORM STATISTICS command 145
 SET TRANCLASS command 194
 SET TRANSACTION command 198
 TRANCLASS, CREATE command 47
 TRANCLASS, DISCARD command 57
 TRANCLASS, INQUIRE command 127
 TRANCLASS, SET command 194
 TRANDUMP
 PERFORM STATISTICS command 145
 TRANDUMPCODE option
 COLLECT STATISTICS command 25
 INQUIRE TRANDUMPCODE command 129
 SET TRANDUMPCODE command 196
 TRANDUMPCODE, INQUIRE command 128
 TRANDUMPCODE, SET command 195
 TRANDUMPING option
 INQUIRE TRANDUMPCODE command 129
 SET TRANDUMPCODE command 196
 TRANISOLATE option
 INQUIRE SYSTEM command 103
 TRANPRIORITY option
 INQUIRE TASK command 108
 TRANSACTION option
 COLLECT STATISTICS command 25
 CREATE TRANSACTION command 50
 DISCARD TRANSACTION command 58
 INQUIRE TASK command 108
 INQUIRE TERMINAL command 122
 INQUIRE TRANSACTION command 133
 PERFORM STATISTICS command 145
 SET TRANSACTION command 198
 transaction security checking 11
 TRANSACTION, DISCARD command 58
 TRANSACTION, INQUIRE command 130
 TRANSACTION, SET command 197
 TRANSID option
 INQUIRE PROGRAM command 90
 INQUIRE REQID command 92
 translator 3
 TRIGGERLEVEL option
 INQUIRE TDQUEUE command 113
 SET TDQUEUE command 181
 TRPROF option
 INQUIRE TASK command 108
 INQUIRE TRANSACTION command 133
 TSQUEUE
 INQUIRE TSQUEUE command 135
 TSQUEUE option
 COLLECT STATISTICS command 25
 PERFORM STATISTICS command 145
 TSQUEUE, INQUIRE command 134
 TTISTATUS option
 INQUIRE TERMINAL command 122
 SET TERMINAL command 187
 TWASIZE option
 INQUIRE TASK command 108
 INQUIRE TRANSACTION command 133
 TYPE option
 INQUIRE FILE command 78
 INQUIRE TDQUEUE command 113
 TYPETERM option
 CREATE TYPETERM command 52
 TYPETERM, CREATE command 51

U

UCTRANST option
 INQUIRE TERMINAL command 122
 SET TERMINAL command 187
 UDSASIZE option
 INQUIRE SYSTEM command 103
 UPDATE option
 INQUIRE FILE command 78
 SET FILE command 162
 USECOUNT option
 INQUIRE PROGRAM command 90
 USERAREA option
 INQUIRE TERMINAL command 123
 USERAREALEN option
 INQUIRE TERMINAL command 123
 USERDATA option
 ACQUIRE TERMINAL command 22
 USERDATALEN option
 ACQUIRE TERMINAL command 22
 USERID option
 INQUIRE REQID command 92
 INQUIRE TASK command 108
 INQUIRE TERMINAL command 123
 USERNAME option
 INQUIRE TERMINAL command 123
 USERSTATUS option
 INQUIRE TRACEFLAG command 126
 SET TRACEFLAG command 192

V

VALIDATIONST option
 INQUIRE TERMINAL command 123
 VALIDITY option
 INQUIRE DSNAME command 70
 VERSION option
 SET PROGRAM command 170
 VFORMST option
 INQUIRE TERMINAL command 123
 VOLUME option
 INQUIRE VOLUME command 137

VOLUME option (*continued*)

SET VOLUME command 199

VOLUME, INQUIRE command 136

VOLUME, SET command 199

VTAM option

COLLECT STATISTICS command 25

PERFORM STATISTICS command 145

VTAM, INQUIRE command 138

VTAM, SET command 200

W

where-clause, CICS command format 5

X

XLNSTATUS option

INQUIRE CONNECTION command 66

XLT option

PERFORM SHUTDOWN command 143

XRFSTATUS option

INQUIRE SYSTEM command 103

Z

ZCPTRACING option

INQUIRE CONNECTION command 67

INQUIRE TERMINAL command 123

SET CONNECTION command 153

SET TERMINAL command 188

Sending your comments to IBM

CICS for MVS/ESA

System Programming Reference

SC33-1171-02

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book. Please limit your comments to the information in this book and the way in which the information is presented.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

You can send your comments to IBM in any of the following ways:

- By mail, use the Readers' Comment Form
- By fax:
 - From outside the U.K., after your international access code use 44 1962 870229
 - From within the U.K., use 01962 870229
- Electronically, use the appropriate network ID:
 - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
 - IBMLink: WINVMD(IDRCF)
 - Internet: idrcf@winvmd.vnet.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The page number or topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Readers' Comments

CICS for MVS/ESA

System Programming Reference

SC33-1171-02

Use this form to tell us what you think about this manual. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer. This form is provided for comments about the information in this manual and the way it is presented.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Be sure to print your name and address below if you would like a reply.

Name

Address

Company or Organization

Telephone

Email



You can send your comments POST FREE on this form from any one of these countries:

Australia	Finland	Iceland	Netherlands	Singapore	United States
Belgium	France	Israel	New Zealand	Spain	of America
Bermuda	Germany	Italy	Norway	Sweden	
Cyprus	Greece	Luxembourg	Portugal	Switzerland	
Denmark	Hong Kong	Monaco	Republic of Ireland	United Arab Emirates	

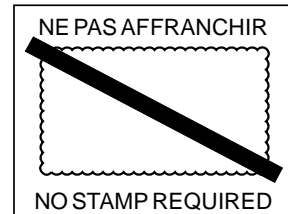
If your country is not listed here, your local IBM representative will be pleased to forward your comments to us. Or you can pay the postage and send the form direct to IBM (this includes mailing in the U.K.).

1 Cut along this line

2 Fold along this line

By air mail
Par avion

IBRS/CCR NUMBER: PHQ-D/1348/SO



**REPONSE PAYEE
GRANDE-BRETAGNE**

IBM United Kingdom Laboratories
Information Development Department (MP095)
Hursley Park,
WINCHESTER, Hants
SO21 2ZZ United Kingdom

3 Fold along this line

From: Name _____
Company or Organization _____
Address _____

EMAIL _____
Telephone _____

1 Cut along this line

4 Fasten here with adhesive tape



Program Number: 5655-018

Printed in U.S.A.

SC33-1171-02



Spine information:



CICS for MVS/ESA

System Programming Reference

Version 4 Release 1