CICS® Transaction Server for OS/390®

CICS System Programming Reference

Release 3

SC33-1689-02

CICS® Transaction Server for OS/390®

CICS System Programming Reference

Release 3

SC33-1689-02

Note!

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

Third edition (March 1999)

This edition applies to Release 3 of CICS Transaction Server for OS/390, program number 5655-147, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

This edition replaces and makes obsolete the previous edition, SC33-1689-01. The technical changes for this edition are summarized under "Summary of changes" and are indicated by a vertical bar to the left of a change.

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, 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.

Contents

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described 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:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply in the United Kingdom or any other 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 NON-INFRINGEMENT, 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.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

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 IBM United Kingdom Laboratories, MP151, 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.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

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

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

ACF/VTAM	DFSMS	MQ
BookManager	DFSMS/MVS	MVS/DFP
C/370	DFSMSdss	MVS/ESA
CICS	DFSMShsm	OS/390
CICS/ESA	IBM	RACF
COBOL/370	IBMLink	SP
DB2	IMS	VTAM

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Preface

What this book is about

This book describes the CICS Transaction Server for OS/390 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 Application Programming Guide*. For information about debugging CICS® applications, see the *CICS 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 Application Programming Reference* manual and the *CICS Application Programming Guide*). Anything that is already documented in those two books is not duplicated here, so you may need to refer to them occasionally.

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

- 'CICS' refers to IBM® CICS Transaction Server for OS/390 Release 3 (called CICS Transaction Server for OS/390 in the rest of this book).
- 'VTAM®' refers to IBM 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, PERFORM, CREATE, and DISCARD commands, together with the DISABLE PROGRAM, ENABLE PROGRAM, EXTRACT EXIT, and RESYNC ENTRYNAME commands and some of the front-end programming interface (FEPI) commands.
- MB equals 1 048 576 bytes.
- KB equals 1024 bytes.

Bibliography

CICS Transaction Server for OS/390

CICS Transaction Server for OS/390: Planning for Installation	GC33-1789
CICS Transaction Server for OS/390 Release Guide	GC34-5352
CICS Transaction Server for OS/390 Migration Guide	GC34-5353
CICS Transaction Server for OS/390 Installation Guide	GC33-1681
CICS Transaction Server for OS/390 Program Directory	GI10-2506
CICS Transaction Server for OS/390 Licensed Program Specification	GC33-1707

CICS books for CICS Transaction Server for OS/390

General	
CICS Master Index	SC33-1704
CICS User's Handbook	SX33-6104
CICS Transaction Server for OS/390 Glossary (softcopy only)	GC33-1705
Administration	
CICS System Definition Guide	SC33-1682
CICS Customization Guide	SC33-1683
CICS Resource Definition Guide	SC33-1684
CICS Operations and Utilities Guide	SC33-1685
CICS Supplied Transactions	SC33-1686
Programming	
CICS Application Programming Guide	SC33-1687
CICS Application Programming Reference	SC33-1688
CICS System Programming Reference	SC33-1689
CICS Front End Programming Interface User's Guide	SC33-1692
CICS C++ OO Class Libraries	SC34-5455
CICS Distributed Transaction Programming Guide	SC33-1691
CICS Business Transaction Services	SC34-5268
Diagnosis	
CICS Problem Determination Guide	GC33-1693
CICS Messages and Codes	GC33-1694
CICS Diagnosis Reference	LY33-6088
CICS Data Areas	LY33-6089
CICS Trace Entries	SC34-5446
CICS Supplementary Data Areas	LY33-6090
Communication	
CICS Intercommunication Guide	SC33-1695
CICS Family: Interproduct Communication	SC33-0824
CICS Family: Communicating from CICS on System/390	SC33-1697
CICS External Interfaces Guide	SC33-1944
CICS Internet Guide	SC34-5445
Special topics	
CICS Recovery and Restart Guide	SC33-1698
CICS Performance Guide	SC33-1699
CICS IMS Database Control Guide	SC33-1700
CICS RACF Security Guide	SC33-1701
CICS Shared Data Tables Guide	SC33-1702
CICS Transaction Affinities Utility Guide	SC33-1777
CICS DB2 Guide	SC33-1939

CICSPlex SM books for CICS Transaction Server for OS/390

General	
CICSPlex SM Master Index	SC33-1812
CICSPlex SM Concepts and Planning	GC33-0786
CICSPlex SM User Interface Guide	SC33-0788
CICSPlex SM View Commands Reference Summary	SX33-6099
Administration and Management	
CICSPlex SM Administration	SC34-5401
CICSPlex SM Operations Views Reference	SC33-0789
CICSPlex SM Monitor Views Reference	SC34-5402
CICSPlex SM Managing Workloads	SC33-1807
CICSPlex SM Managing Resource Usage	SC33-1808
CICSPlex SM Managing Business Applications	SC33-1809
Programming	
CICSPlex SM Application Programming Guide	SC34-5457
CICSPlex SM Application Programming Reference	SC34-5458
Diagnosis	
CICSPlex SM Resource Tables Reference	SC33-1220
CICSPlex SM Messages and Codes	GC33-0790
CICSPlex SM Problem Determination	GC33-0791

Other CICS books

CICS Application Programming Primer (VS COBOL II)	SC33-0674
CICS Application Migration Aid Guide	SC33-0768
CICS Family: API Structure	SC33-1007
CICS Family: Client/Server Programming	SC33-1435
CICS Family: General Information	GC33-0155
CICS 4.1 Sample Applications Guide	SC33-1173
CICS/ESA 3.3 XRF Guide	SC33-0661

If you have any questions about the CICS Transaction Server for OS/390 library, see *CICS Transaction Server for OS/390: Planning for Installation* which discusses both hardcopy and softcopy books and the ways that the books can be ordered.

Books from related libraries

You may also need to refer to relevant MVS publications.

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
- *OS/390 Security Server (RACF) Security Administrator's Guide*, SC28-1915
- *DFSMS/MVS DFSMSdfp Advanced Services*, SC26-4921
- *OS/390 MVS Diagnosis: Procedures*, SY28-1082

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 usually in step. However, due to the time required to print and distribute hardcopy books, the BookManager version is more likely to have had last-minute changes made to it before publication.

Subsequent updates will probably be available in softcopy before they are available in hardcopy. This means that at any time from the availability of a release, softcopy versions should be regarded as the most up-to-date.

For CICS Transaction Server 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.

Updates to the softcopy are clearly marked by revision codes (usually a “#” character) to the left of the changes.

Summary of changes

Changes for this edition of the manual are marked by vertical bars to the left of the text.

Changes for CICS Transaction Server for OS/390 release 3

This edition is based on the previous CICS Transaction Server for OS/390 edition, SC33-1689-01.

For CICS Transaction Server for OS/390 release 3, the following changes have been made:

- The system programming interface commands:
  - EXEC CICS CREATE
  - EXEC CICS DISCARD
  - EXEC CICS INQUIRE
  - EXEC CICS SET

  are enhanced to provide function for the management of the following new resources:
  - DOCTEMPLATE
  - ENQMODEL
  - PROCESSTYPE
  - REQUESTMODEL
  - TCPIPSERVICE
  - TSMODEL
- The EXEC CICS INQUIRE TDQUEUE command is extended to return the name of the MEMBER when the DDNAME references a partitioned data set.
- The system programming interface commands:
  - EXEC CICS CREATE
  - EXEC CICS DISCARD
  - EXEC CICS INQUIRE
  - EXEC CICS SET

  are enhanced to provide function for the management of programs running under a JVM.

Changes for CICS Transaction Server for OS/390 release 2

For CICS Transaction Server for OS/390 release 2, the following changes have been made:

- Addition of BRIDGE and IDENTIFIER to the INQUIRE TASK command to support the 3270 bridge.
- Addition of BREXIT and FACILITYLIKE to the INQUIRE TRANSACTION command to support the 3270 bridge.
- The CICS DB2 attachment facility is enhanced to provide resource definition online (RDO) support for DB2 resources as an alternative to resource control table (RCT) definitions. The system programming interface is enhanced to provide function for the management of the CICS DB2 attachment facility. The following new commands are added for DB2 resource definitions, DB2CONN, DB2ENTRY, and DB2TRAN:
  - EXEC CICS CREATE
  - EXEC CICS DISCARD
  - EXEC CICS INQUIRE
  - EXEC CICS SET

  The following commands are extended:
  - EXEC CICS INQUIRE TASK returns the name of the DB2 plan used for the task.
  - EXEC CICS INQUIRE SYSTEM returns the name of the DB2CONN currently installed.

Changes for CICS Transaction Server for OS/390 release 1

For CICS Transaction Server for OS/390, the following changes have been made:

- New DISCARD CONNECTION, DISCARD JOURNALMODEL, DISCARD JOURNALNAME, DISCARD TDQUEUE, DISCARD TERMINAL, INQUIRE JOURNALMODEL, INQUIRE JOURNALNAME (replacing INQUIRE JOURNALNUM), INQUIRE STREAMNAME, INQUIRE UOW, INQUIRE UOWDSNFAIL, INQUIRE UOWENQ, INQUIRE UOWLINK, PERFORM ENDAFFINITY, SET JOURNALNAME (replacing SET JOURNALNUM), SET UOW, and SET UOWLINK commands
- New or changed options for COLLECT STATISTICS, DISABLE PROGRAM, ENABLE PROGRAM, INQUIRE CONNECTION, INQUIRE DSNAME, INQUIRE EXITPROGRAM, INQUIRE FILE, INQUIRE NETNAME, INQUIRE SYSTEM, INQUIRE TASK, INQUIRE TDQUEUE, INQUIRE TERMINAL, INQUIRE TRANSACTION, INQUIRE TSQUEUE, PERFORM SHUTDOWN, PERFORM STATISTICS RECORD, RESYNC ENTRYNAME, SET CONNECTION, SET DSNAME, and SET FILE commands
- Addition of the CREATE commands
- INQUIRE VOLUME and SET VOLUME commands now obsolete
- Removal of local DL/I options
- New conditions on the SET TDQUEUE command

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

Changes for the CICS/ESA 4.1 edition

The main changes for CICS/ESA 4.1 are summarized below:

- Transfer of the SPOOLCLOSE, SPOOLOPEN INPUT, SPOOLOPEN OUTPUT, SPOOLREAD, and SPOOLWRITE command descriptions to the *CICS 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 DELETSHIPPED, PERFORM DELETESHIPPED, and SET DELETSHIPPED 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, GMMLENGTH, GMMTEXT, STARTUP, and STARTUPDATE options to the INQUIRE SYSTEM command
- Addition of ASCII, AUTOCONNECT, DATASTREAM, DEVICEST, and POOLTERMID options to the INQUIRE TERMINAL command
- Addition of GMMLENGTH 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 were made to clarify the information in this manual.

Chapter 1. Introduction

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 Application Programming Reference* manual. A third manual, the *CICS 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 (EDF), 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 SYSID option is provided, are an exception.

---

[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.

- Additional security checking is available for SPI commands, as explained on page 2.
- Programs containing SPI commands must be translated with the SP translator option, as explained in “Security checking” on page 14.

There are also special considerations that apply to certain groups of commands. These notes begin on page 17.

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
- 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 13.

Chapter 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 14.

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 Application Programming Guide* for translator options, and the *CICS System Definition Guide* for the job control language.

CICS syntax notation

Throughout this book, the syntax for each command is presented in the form of a diagram. 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

These have the same meaning in SPI commands as in API commands. (See the *CICS Application Programming Guide* for basic information about these options, and “Exception conditions” on page 13 for additional SPI details.)

You read the diagram by following the arrows from left to right, using these conventions:

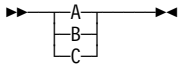
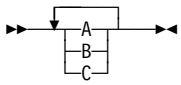
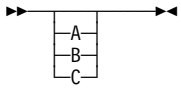
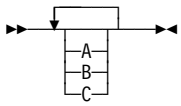
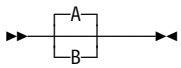
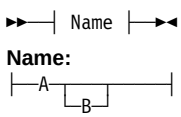
Symbol	Action
	A set of alternatives—one of which you ***must*** code.
	A set of alternatives—one of which you ***must*** code. You ***may*** code more than one of them, in any sequence.
	A set of alternatives—one of which you ***may*** code.
	A set of alternatives — any number (including none) of which 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).

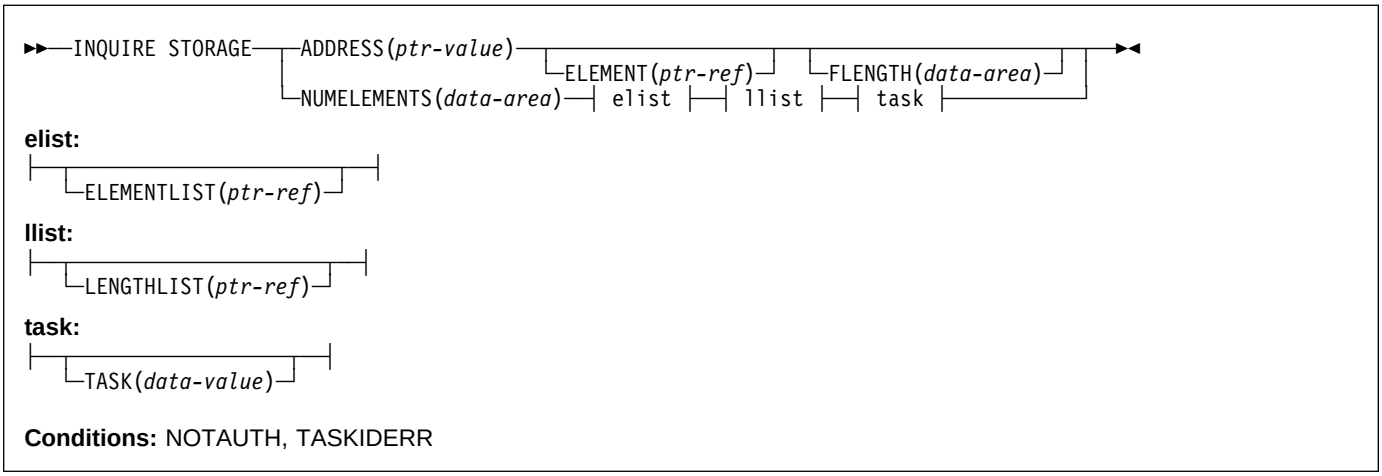


Figure 1. Syntax diagram for the INQUIRE STORAGE command

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 2).
- 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 10.
- 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 executing the same program from interfering with one another, CICS keeps a separate copy of

program areas that may change during execution (sometimes called “working storage”) for each task. This means that any area that may be modified, including data-area arguments to CICS commands, must reside either in such an area of the program or in storage outside the program which the application design allows the program to modify.

Some of this storage is allocated automatically; this category includes the WORKING-STORAGE section in COBOL programs, AUTOMATIC storage in PL/I and C/370, and areas appended to the DFHEISTG DSECT in assembler. It can also be allocated explicitly with a CICS GETMAIN command or a language facility such as a PL/I ALLOCATE statement, in this or a preceding program. This category includes the LINKAGE section in COBOL, BASED and CONTROLLED storage in PL/I, and other DSECTs in assembler. See the *CICS Application Programming Guide* for more detail about CICS re-entrancy rules.

- The program that issues the command has *write* access to the area. CICS changes the content of data-areas and, therefore, you cannot use storage that you are not allowed to change.

  Write access is affected by the storage protection key in which the program is running, and by the transaction isolation status of its task. See the discussion of these subjects in the *CICS Application Programming Guide* and the *CICS System Definition Guide*, and the TRANISOLATE option of a TRANSACTION definition in the *CICS Resource Definition Guide*.

- The MVS/ESA™ restrictions on addressing mode that apply to all CICS commands are observed. These are enforced automatically in high-level languages but, in assembler, the program must be in primary addressing mode, and the primary address space must be the home address space. All arguments for options must reside in the primary address space.

  Note: CICS does not always preserve access registers across CICS commands or macro invocations. If your program uses access registers, it should save them before invoking a CICS service, and restore them before reusing them.

Any area that can be used for a data-area can also be used for a data-value. In addition, you can use areas not allowed for data-areas, because CICS never changes a data-value. In particular, you can use:

- Constants, including literals. In the example above, for instance, you could use a literal instead of a variable for the program name:

  ```
  EXEC CICS SET TDQUEUE ('TAX')
            TRIGGERLEVEL(1) END-EXEC
  ```

  When you use a numeric literal in a command, the translator ensures a constant of the correct type and length, provided the literal is capable of being converted to such a constant, as in TRIGGERLEVEL above. In COBOL and assembler, the translator also ensures character literals of the correct length, padding with blanks if the literal is shorter than the length the argument requires. In C/370 and PL/I, however, you must do this yourself:

  ```
  EXEC CICS SET TDQUEUE ('TAX ')
            TRIGGERLEVEL(1);
  ```

- Other program areas not in “working storage,” such as static storage in PL/I.
- Areas to which your program has read but not write access (the link-pack area, for example).

Note: Sometimes an option is used both to send and receive information, although this usage occurs more often in API than SPI commands. When it does, the argument must be a data-area, because CICS stores into it.

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 Application Programming Reference* manual 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.
```

Appendix A, “CICS-value data areas used by all commands” on page 321 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 null values, as explained in “SET commands” on page 21.
- 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.
```

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;
 }
```

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;
```

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
```

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 relevant language manual for information, although some language-specific information may be found in the *CICS Application Programming Guide*.

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 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.

In COBOL, VS COBOL II, PL/I and assembler language, if the translator option NOLENGTH is used, the translator does 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 17 and “SET commands” on page 21 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 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 335.

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. Appendix B, “EXEC interface block (EIB) response and function codes” on page 335 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 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 System Definition Guide* and the *CICS Resource Definition Guide*, respectively, and the *CICS Customization Guide* contains general information on CICS security.

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

Transaction
Command
Surrogate
Resource

Authtype (DB2® objects only)

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 option in the TRANSACTION is YES or the CMDSEC system initialization parameter 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 2).

If the command associates a user ID with a resource, a **surrogate** check may follow the command check. This check ensures that the user ID of the task issuing the command has authority to act as a surrogate for the user ID 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
SET DB2CONN with AUTHID or COMAUTHID
SET DB2ENTRY with AUTHID
SET DB2TRAN that references a DB2ENTRY containing AUTHID
CREATE CONNECTION with SECURITYNAME
CREATE DB2CONN with AUTHID or COMAUTHID
CREATE DB2ENTRY with AUTHID
CREATE DB2TRAN that references a DB2ENTRY containing AUTHID
CREATE SESSIONS with USERID
CREATE TDQUEUE 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 **resource** check verifies that the user ID 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, and exits
XTST for temporary storage queues
XDB2 for DB2 entries and transactions

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 20 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 JOURNALNAME (R) COLLECT STATISTICS JOURNALNUM (R) DISCARD JOURNALNAME (A) INQUIRE JOURNALNAME (R) SET JOURNALNAME (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) INQUIRE TSQNAME (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) CREATE TDQUEUE (A) DISCARD TDQUEUE (A) INQUIRE TDQUEUE (R) SET TDQUEUE (U)
DB2ENTRYs (XDB2 option)	CREATE DB2ENTRY (A) CREATE DB2TRAN (A) INQUIRE DB2ENTRY (R) INQUIRE DB2TRAN (R) SET DB2ENTRY (U) SET DB2TRAN (U)

Resource (system initialization parameter)	Commands
DB2TRANs (XDB2 option)	CREATE DB2ENTRY (A) CREATE DB2TRAN (A) INQUIRE DB2ENTRY (R) INQUIRE DB2TRAN (R) SET DB2ENTRY (U) SET DB2TRAN (U)

Authtype checking applies to DB2CONNs, DB2ENTRYs, and DB2TRANs only. For more information, see the *CICS DB2 Guide*.

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 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)
Coupling facility data table server connections (CFDTPOOL)
Connections (CONNECTION and UOWLINK)
DB2 connections (DB2CONN)
DB2 entries (DB2ENTRY)
DB2 transactions (DB2TRAN)
DOC templates (DOCTEMPLATE)
Exits (EXITPROGRAM)
External data sets (DSNAME and UOWDSNFAIL)
Files (FILE)
Journals (JOURNALNAME and JOURNALMODEL)
Log streams (STREAMNAME)
Map sets (PROGRAM)
Partition sets (PROGRAM)
Partners (PARTNER)
Profiles (PROFILE)
Programs (PROGRAM)
Session groups (MODENAME)
System dump codes (SYSDUMPCODE)
TCP/IP services (TCPIPSERVICE)
Temporary storage queues (TSQUEUE)
Temporary storage pools (TSPOOL)
Temporary storage models (TSMODEL)
Terminals (TERMINAL, NETNAME)
Transaction classes (TCLASS, TRANCLASS)
Transaction dump codes (TRANDUMPCODE)
Transactions (TRANSACTION)
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 Resource Definition Guide*.

The system elements about which you can inquire are:

Autoinstall for terminals (AUTOINSTALL)
Dump data sets (DUMPDS)
Enqueues (UOWENQ)
Interregion communication (IRC)
Monitor (MONITOR)
Requests (REQID)
Shipped terminal status (DELETSHIPPED)
Statistics (STATISTICS)
Storage (STORAGE)
System status (SYSTEM)
Tasks (TASK, TASK LIST)
TCP/IP (TCPIP)
Tracing (TRACEDEST, TRACEFLAG, TRACETYPE)
Units of work (UOW, UOWDSNFAIL, UOWENQ, UOWLINK)
VTAM

Most of these elements correspond to system initialization parameters. If you need more information about them, see the system initialization parameters discussion in the *CICS 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 13 . (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” on page 17, plus requests (REQIDs) and units of work (UOWs, UOWDSNFAILs, UOWENQs, and UOWLINKs).

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:

Browse START

```
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 and INQUIRE UOWENQ are exceptions 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:

- DB2ENTRYs and DB2TRANs
- 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:

START browse AT

```
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 restricts 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:

Browse NEXT

```
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:

Browse END

```
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 RESPCODE = DFHRESP(END)
    EXEC CICS INQUIRE FILE(FILENAME) NEXT
        OPENSTATUS(OPENSTAT)
        RECOVSTAT(RCVRSTAT)
        FWDRECSTATUS(FWDSTAT)
        RESP(RESPCODE) END-EXEC
    IF RESPCODE = 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 giving up 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) DSNAME(DSN) END-EXEC.
```

See “Null values” on page 13 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
- DB2 connection
- DB2 resources (DB2ENTRYs DB2TRANs)
- Document templates
- ENQ models
- Files
- Journal models
- LSR pools
- Map sets
- Partition sets
- Partners
- Process types
- Profiles
- Programs
- Request models
- Sessions
- TCP/IP service
- Temporary storage queue models
- Transient data queues
- 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 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, 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 Resource Definition Guide* for specific rules about the argument values.

- You can use one or more blanks to separate attributes 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 similar 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, 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 use defaults for 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.
- You can omit the ATTRLEN option 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 12.

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 341 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
DB2Conns
DB2Entrys
DB2Trans
| Document templates
| ENQ models
Files
Journals and journal models
Partners
| Process types
Profiles

Programs, map sets, and partition sets
Request models
TCP/IP service
Temporary storage queue models
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 23 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 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.

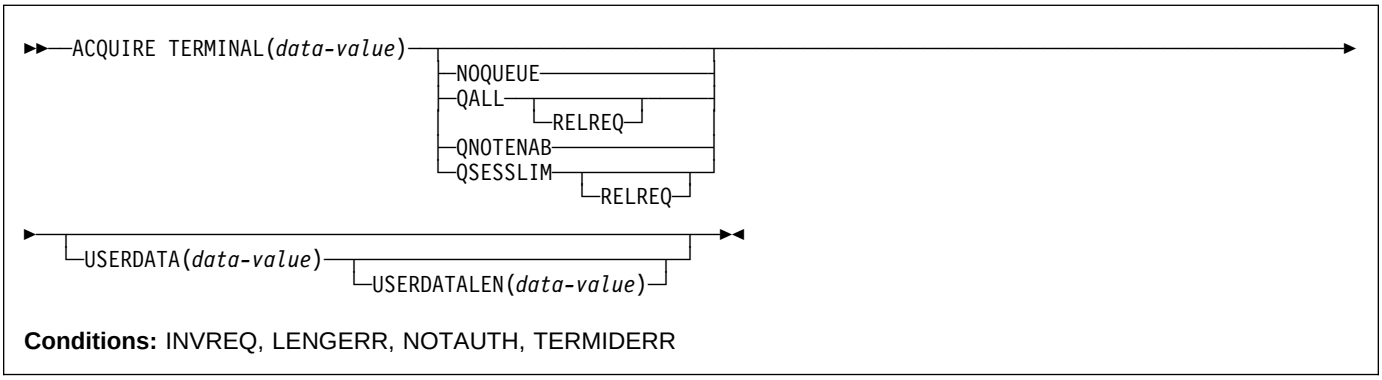
Because of this default (and some history), CICS requires that you always identify an exit in the same way that you did in the ENABLE PROGRAM command that created it—that is, by coding (or omitting) the same PROGRAM and ENTRYNAME values. RESYNC ENTRYNAME is an exception; you specify the exit name in 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.

Chapter 2. System commands

ACQUIRE TERMINAL

Acquire a session with a terminal.



Conditions: INVREQ, LENGERR, NOTAUTH, TERMIDERR

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 reestablished 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 reestablished.

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.

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 Application Programming Reference* manual 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 values:

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

NOTAUTH
RESP2 values:

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

TERMIDERR
RESP2 values:

1 The terminal cannot be found.

COLLECT STATISTICS

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

Conditions: INVREQ, IOERR, NOTAUTH, NOTFND

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 33 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.

Table 1. Resource types and statistics

Resource type	Statistic type	Global statistics	Specific statistics
AUTOINSTALL	Terminal autoinstall	DFHA04DS	-
CONNECTION	ISC/IRC system and mode entries	-	DFHA14DS
DB2CONN	DB2 Connection	DFHD2GDS	-
DB2ENTRY	DB2 Entry	-	DFHD2RDS
DISPATCHER	Dispatcher	DFHDSGDS	-
ENQUEUE	Enqueue	DFHNQGDS	-
FEPI CONNECTION	FEPI Connection	-	DFHA23DS
FEPI POOL	FEPI Pool	-	DFHA22DS
FEPI TARGET	FEPI Target	-	DFHA24DS
FILE	File control	-	DFHA17DS
JOURNALNAME	Journalname	-	DFHLGRDS
JOURNALNUM	Journalname	-	DFHLGRDS
LSRPOOL	LSR pools	-	DFHA08DS
MONITOR	Monitor	DFHMNGDS	DFHMNTDS
PROGAUTO	Program autoinstall	DFHPGGDS	-
PROGRAM	Program	DFHLDGDS	DFHLDRDS
RECOVERY	Recovery manager	DFHRMGDS	-
STORAGE	Storage manager	DFHSMSDS	DFHSMDDS
STREAMNAME	Log stream	-	DFHLGSDS
SYSDUMPCODE	Dump (system)	DFHSDGDS	DFHSDRDS
TABLEMGR	Table manager	DFHA16DS	-
TCLASS	Transaction class	-	DFHXMCDS
TCPIPSERVICE	TCP/IP service	-	DFHSORDS
TDQUEUE	Transient data	DFHTQGDS	DFHTQRDS
TERMINAL	Terminals	-	DFHA06DS
TRANCLASS	Transaction class	-	DFHXMCDS
TRANDUMPCODE	Dump (transaction)	DFHTDGDS	DFHTDRDS
TRANSACTION	Transaction manager	DFHXMGDS	DFHXMRDS
TSQUEUE	Temporary storage	DFHTSGDS	
VTAM	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	CICSTS13.CICS.SDFHMAC
COBOL	CICSTS13.CICS.SDFHCOB
PL/I	CICSTS13.CICS.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 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.

DB2CONN
requests statistics for the CICS DB2 connection including information for pool threads and command threads.

DB2ENTRY(*data-value***)**
requests statistics for a DB2ENTRY; data-value is the 8-character identifier of the DB2ENTRY (from its DB2ENTRY definition).

DISPATCHER
requests global statistics on the dispatcher domain.

ENQUEUE
requests global statistics for enqueue requests.

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

JOURNALNAME(*data-value***)**
requests statistics for a CICS journal; data-value is an 8-character journal name. CICS returns the address of the area of storage that contains the requested statistics.

To collect statistics for journals defined using the journal numbering convention (for example, for the auto journals defined in file resource definitions), specify the name as DFH*nn*, where nn is the journal number in the range 01 to 99.

Note: Specifying DFHJ01 returns statistics written to a user journal of that name, *not* the system log.

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. CICS returns the address of the area of storage that contains the requested statistics.

Specifying JOURNALNUM(1) returns statistics for journal DFHJ01. Note that this is *not* the system log.

Specifying identifiers in the range 1—99 returns statistics for journals DFHJ01—DFHJ99.

Note: JOURNALNUM continues to be supported for compatibility with releases of CICS earlier than CICS Transaction Server for OS/390. However, the statistics returned are CICS log manager statistics, *not* journal control statistics. You can map the data at the address returned only by using the DFHLGRDS DSECT which replaces the DFHA13DS DSECT supported at earlier releases.

When you make changes to application programs that use JOURNALNUM, you are recommended to use the JOURNALNAME option.

LASTRESET(*data-area***)**
returns a 4-byte packed decimal field 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 0hhmmss+), 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 Application Programming Reference* manual.) 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 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.

RECOVERY
requests global statistics on the recovery manager.

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 Performance Guide*. Without a data-value, this option requests the global statistics for the CICS dynamic storage areas.

STREAMNAME(*data-value***)**
requests statistics for a logstream; data-value is the 26-character name of the logstream.

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 1-10, 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')).

TCPIPSERVICE(*data-value***)**
requests the statistics for a TCP/IP service; data-value is the 8-character name of the TCP/IP service.

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. 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 values:

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

IOERR
RESP2 values:

3 The requested statistics area was not functioning. This happens 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 JOURNALNAME you specify is not known to CICS).
2 The type of resource is not defined in the CICS system (for example, FEPI statistics are requested with POOL or NODE when the FEPI system initialization parameter specifies NO).

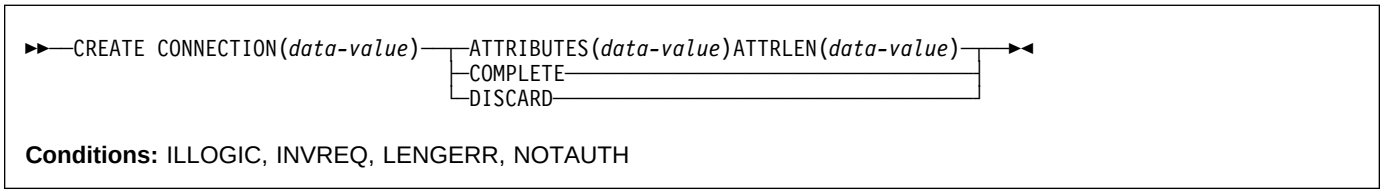
Examples

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 Transaction Server for OS/390 to produce information about a CICS system. The reports include a CICS and MVS storage analysis that can be used as an aid to specifying the DSA LIMIT parameters.

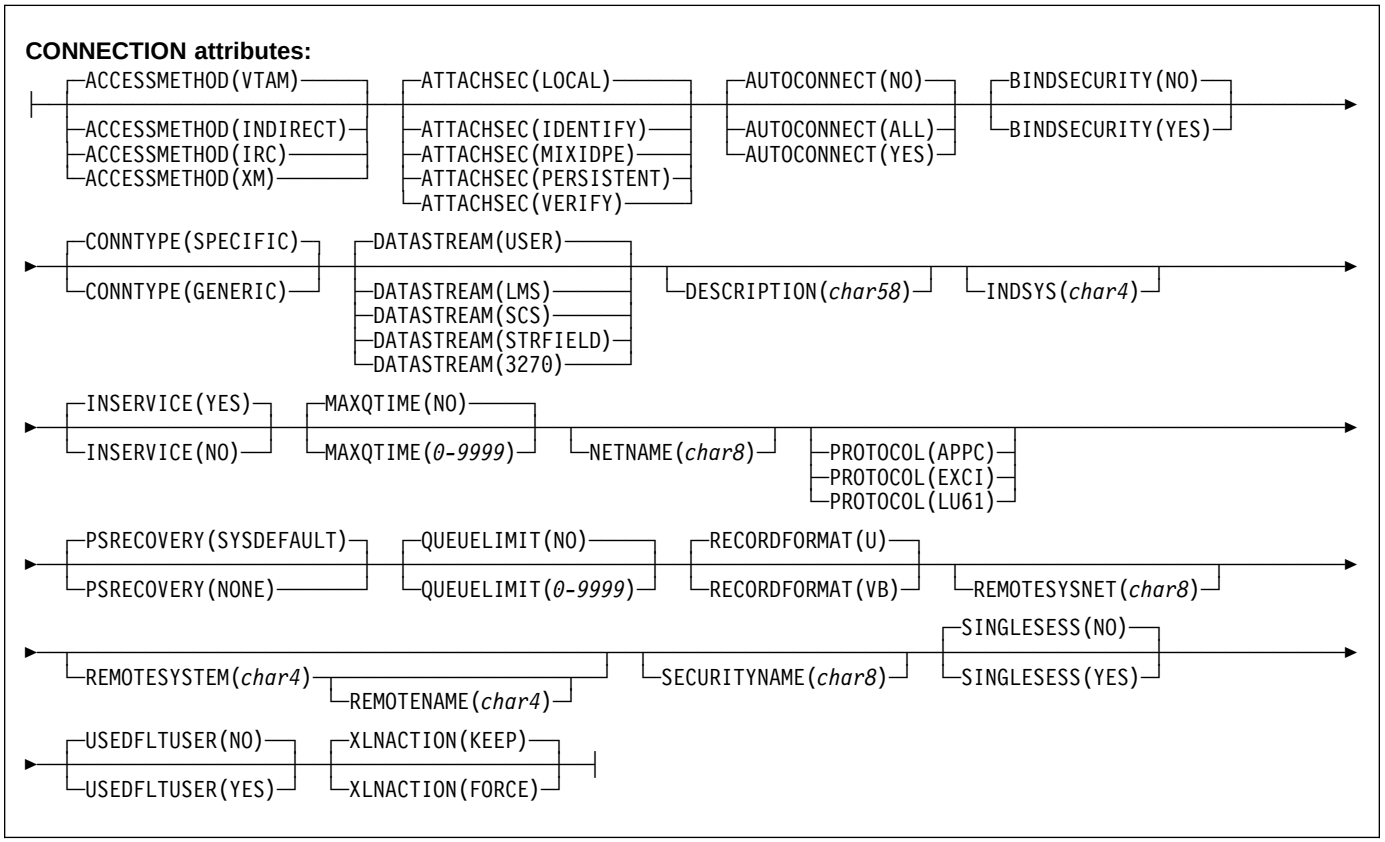
See the *CICS Performance Guide* for information on installing and operating the DFH0STAT application. The source code for the application can be found in CICSTS13.CICS.SDFHSAMP.

CREATE CONNECTION

Define a CONNECTION in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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 22 for other general rules about CREATE commands.

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

1. CREATE CONNECTION with the ATTRIBUTES and ATTRLEN options
2. CREATE SESSIONS
3. Additional CREATE SESSIONS if desired (only one group of sessions is required, but you can define additional groups)
4. 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 discards 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 23 for general rules for specifying attributes, and the CONNECTION chapter in the *CICS 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 is not to 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 syntax 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 341 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 DB2CONN

Define a DB2CONN in the local system.

Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE DB2CONN command builds a DB2CONN definition, without reference to data stored in the CSD file. If there is already a DB2CONN 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 DB2CONN 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.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the DB2CONN being added. The list of attributes must be coded as a single character string using the syntax shown in DB2CONN attributes.

You can assign default values for all attributes of a DB2CONN definition by specifying an ATTRLEN value of

0. You still need to specify the ATTTRIBUTES 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.

DB2CONN(*data-value***)**
specifies the 8-character name of the DB2CONN 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 syntax 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 341 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:

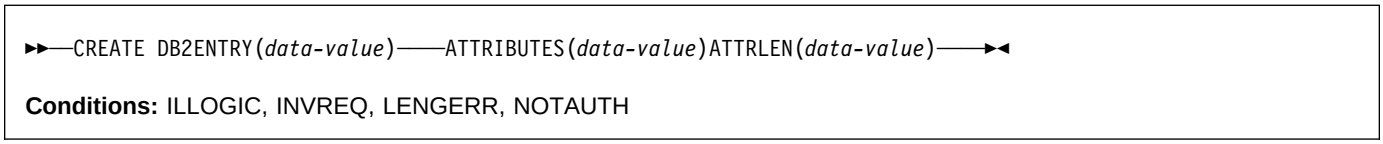
100 The user associated with the issuing task is not authorized to use this command.

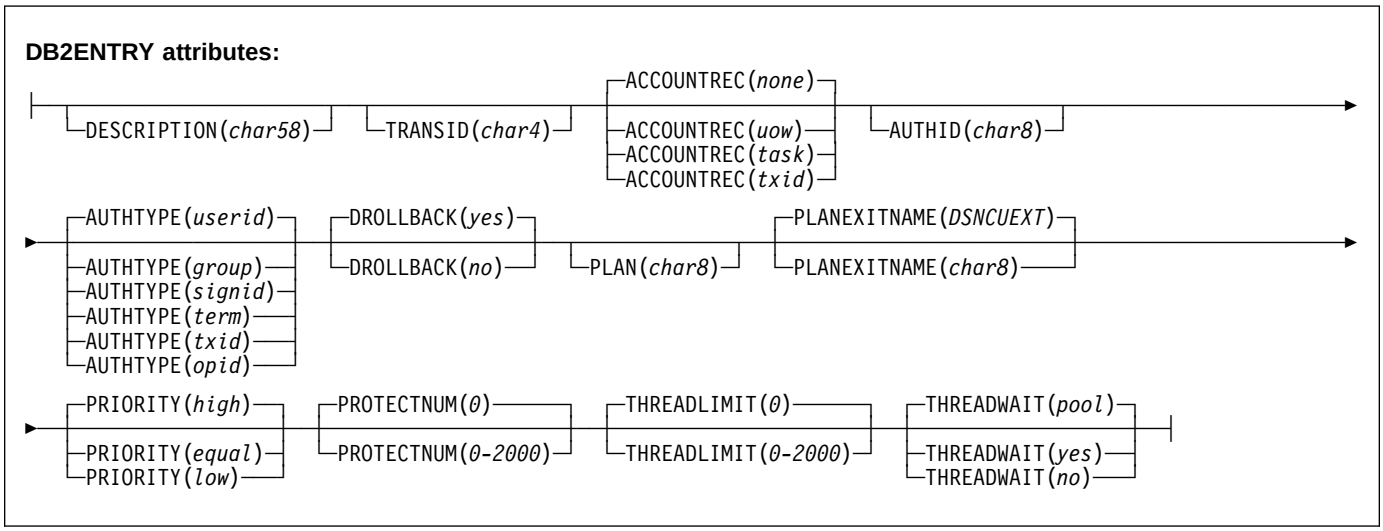
102 The user associated with the issuing task is not an authorized surrogate of the user specified in the AUTHID, COMAUTHID, or SIGNID parameter.

103 The user associated with the issuing task is not authorized to create a DB2CONN with an AUTHTYPE or COMAUTHTYPE parameter.

CREATE DB2ENTRY

Define a DB2ENTRY in the local system.





Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE DB2ENTRY command builds a DB2ENTRY definition, without reference to data stored in the CSD file. If there is already a DB2ENTRY with the name you specify in the local CICS region, the command fails unless the existing DB2ENTRY is disabled, in which case the new definition replaces the old one. If no DB2ENTRY with the name specified exists, the new definition is added.

A syncpoint is implicit in CREATE DB2ENTRY 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.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the DB2ENTRY being added. The list of attributes must be coded as a single character string using the syntax shown in DB2ENTRY attributes.

You can assign default values for all attributes of a DB2ENTRY definition by specifying an ATTRLEN value of 0. You still need to specify the ATTTRIBUTES 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.

DB2ENTRY(*data-value***)**
specifies the 8-character name of the DB2ENTRY 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 syntax 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 341 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.

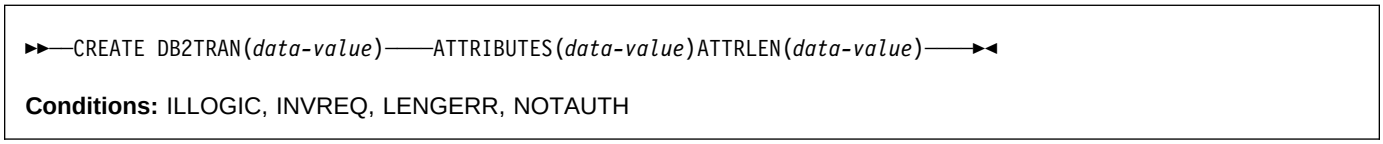
101 The user associated with the issuing task is not authorized to create a DB2ENTRY definition with this name.

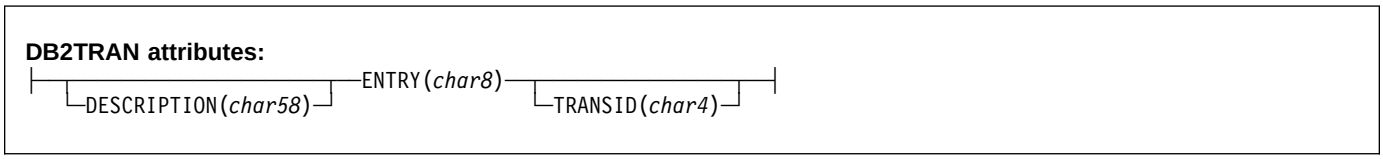
102 The user associated with the issuing task is not an authorized user specified in the AUTHID parameter.

103 The user associated with the issuing task is not authorized to create this DB2ENTRY with an AUTHTYPE parameter.

CREATE DB2TRAN

Define a DB2TRAN in the local system.





Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE DB2TRAN command builds a DB2TRAN definition, without reference to data stored in the CSD file. If there is already a DB2TRAN 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 DB2TRAN 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.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the DB2TRAN being added. The list of attributes must be coded as a single character string using the syntax shown in DB2TRAN attributes.

You can assign default values for all attributes of a DB2TRAN definition by specifying an ATTRLEN value of 0. You still need to specify the ATTTRIBUTES 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.

DB2TRAN(*data-value***)**
specifies the 8-character name of the DB2TRAN 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 syntax 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 341 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:

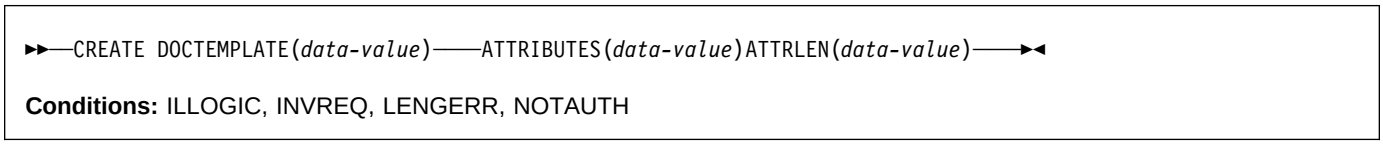
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to create a DB2TRAN definition and associate it with the names DB2ENTRY.

102 The user associated with the issuing task is not an authorized surrogate of the user specified in the AUTHID parameter of the DB2ENTRY named in the DB2TRAN.

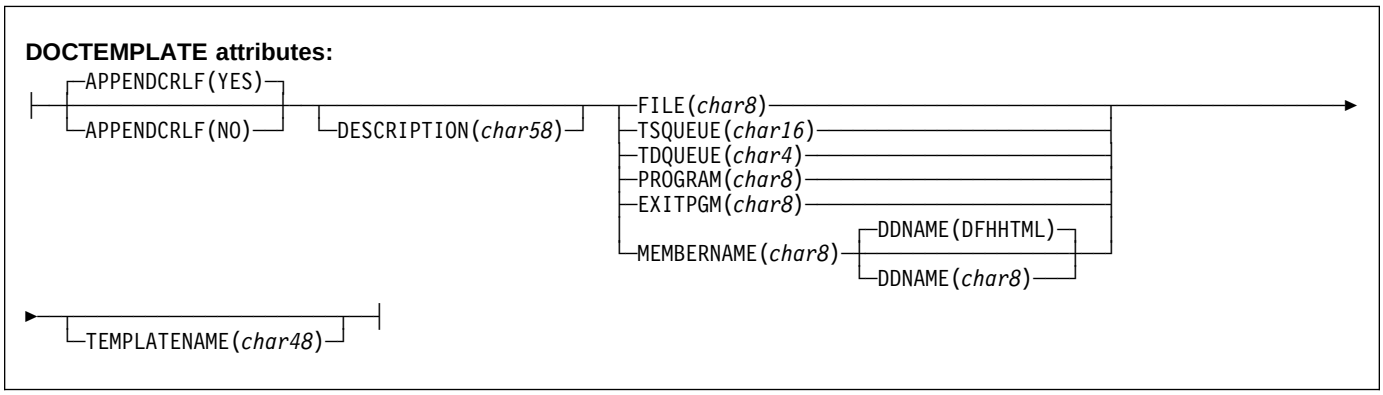
103 The user associated with the issuing task is not authorized to associate this DB2TRAN with the names DB2ENTRY specifying AUTHTYPE.

CREATE DOCTEMPLATE

Define a document template.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE DOCTEMPLATE command builds a document template definition in the local CICS region, without reference to data stored on the CSD file. If there is already a document template with 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 DOCTEMPLATE 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the template being added. The list of attributes must be coded as a single character string using the syntax shown in DOCTEMPLATE attributes. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the DOCTEMPLATE chapter in the *CICS 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.

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

Conditions

INVREQ
RESP2 values:

n There is a syntax 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 341 for information on RESP2 values.

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

LENGERR
RESP2 values:

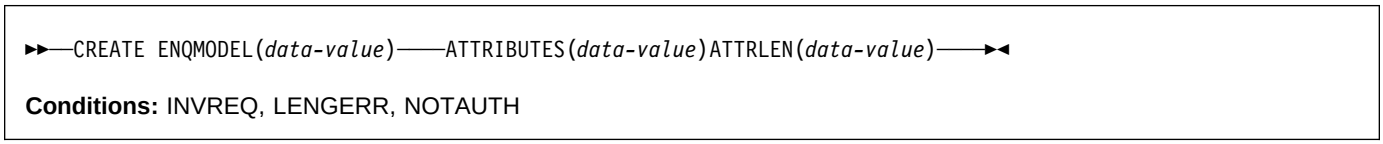
1 The length you have specified in ATTRLEN is negative.

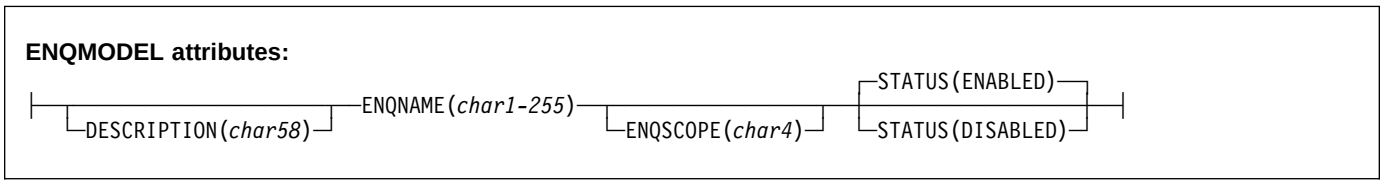
NOTAUTH
RESP2 values:

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

CREATE ENQMODEL

Define an ENQMODEL resource definition.





Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE ENQMODEL command builds an enqueue model resource definition in the local CICS region, without reference to data stored on the CSD file. If there is already an ENQMODEL with the name you specify in the local CICS region, the new definition replaces the old one; if not, the new definition is added.

When CREATE is issued, the ENQMODEL is put into the WAITING state until there are no enqueues in the local system which match the ENQNAME pattern. It is then ENABLED or DISABLED, as specified in the CREATE command.

ENQMODELs forming nested generic enqnames must either be installed in the disabled state, or be installed in order, from the most to the least specific. If another ENQMODEL with the same or a less specific nested enqname is already installed enabled, INVREQ is returned to the caller.

For example: If an ENQMODEL containing AB* is installed, it must be discarded or disabled before creating an ENQMODEL with ABCD*.

A syncpoint is implicit in CREATE ENQMODEL 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the ENQMODEL being added. The list of attributes must be coded as a single character string using the syntax shown in **ENQMODEL attributes**. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the ENQMODEL chapter in the *CICS 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.

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

Conditions

INVREQ
RESP2 values:

n There is a syntax 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 341 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:

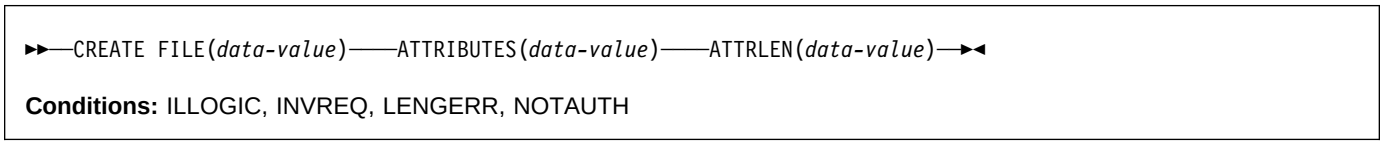
1 The length you have specified in ATTRLEN is negative.

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 an ENQMODEL definition with this name.

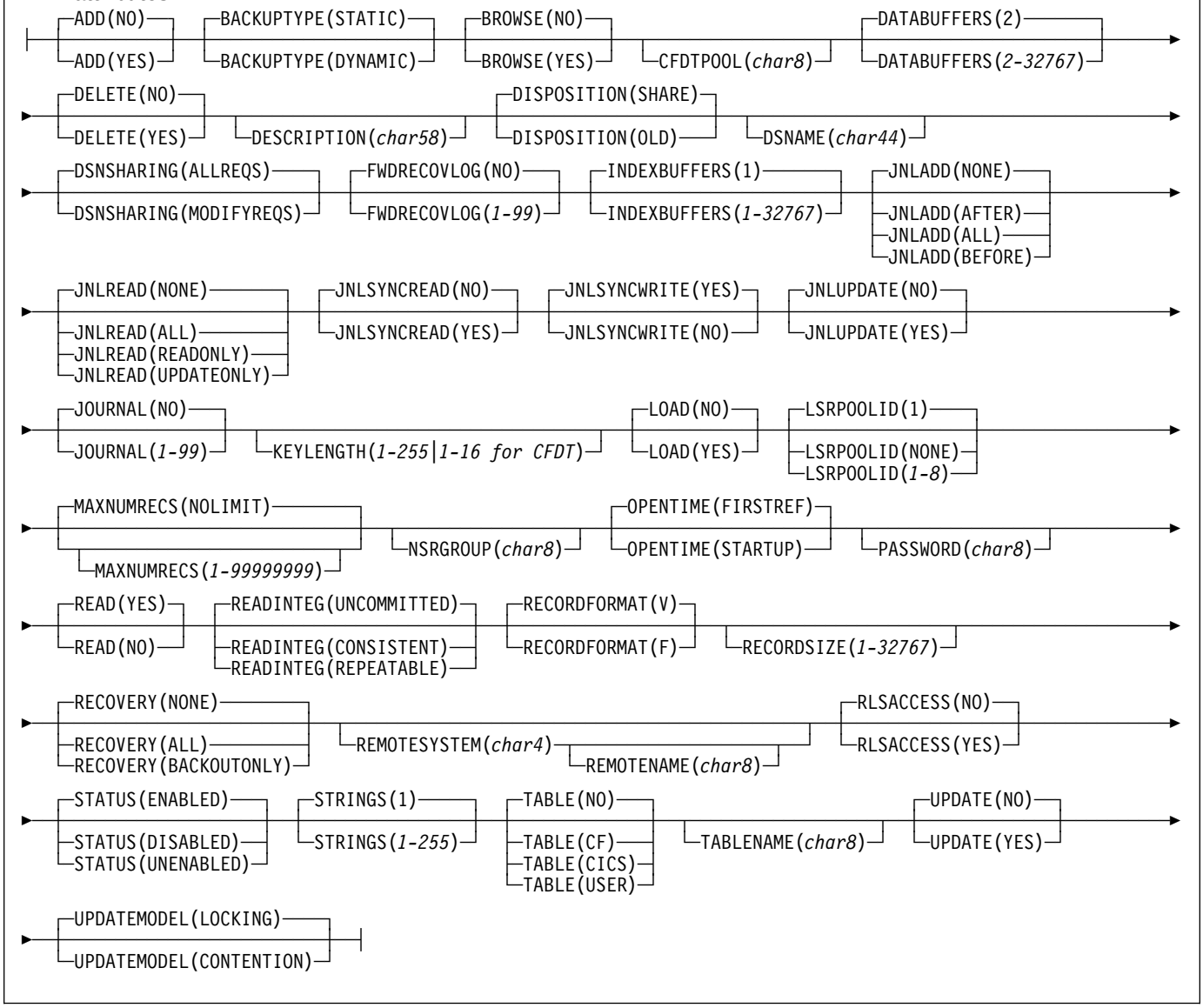
CREATE FILE

Define a FILE in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

FILE attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE FILE command builds a FILE definition, without reference to data stored on the CSD file. If there is already a file with 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 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 22 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 23 for general rules for specifying attributes, and the FILE chapter in the *CICS 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 syntax 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 341 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.

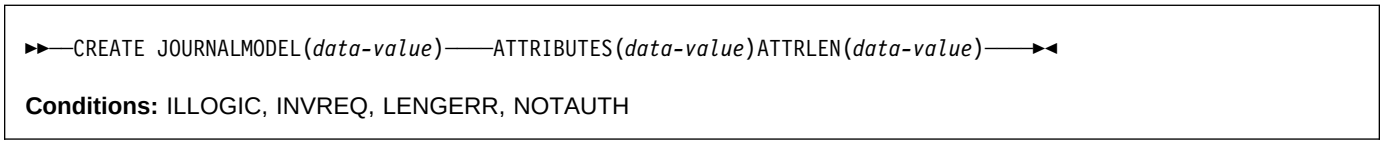
NOTAUTH
RESP2 values:

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

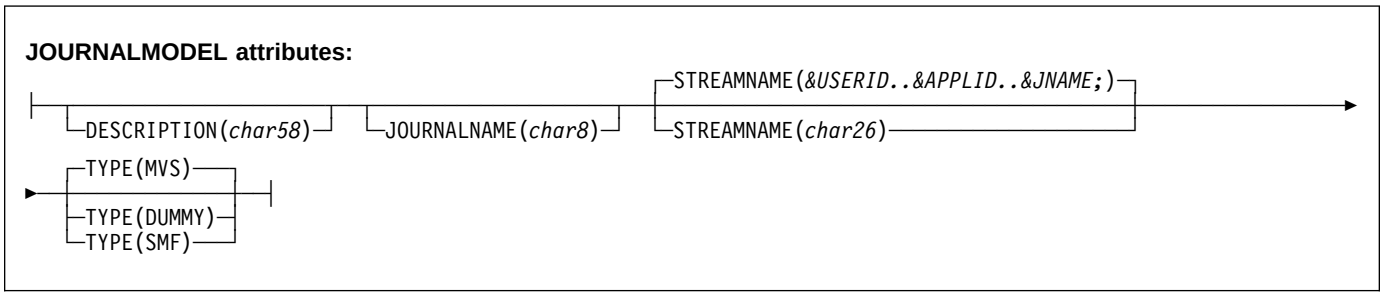
101 The user associated with the issuing task is not authorized to create a FILE definition with this name.

CREATE JOURNALMODEL

Define a journal model in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE JOURNALMODEL command builds a JOURNALMODEL definition, without reference to data stored on the CSD file. If there is already a journal model with the name you specify in the local CICS region, the new definition replaces the old one; it not, the new definition is added.

A syncpoint is implicit in CREATE JOURNALMODEL 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the JOURNALMODEL being added. The list of attributes must be coded as a single character string using the syntax shown in **JOURNALMODEL attributes**. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the JOURNALMODEL chapter in the *CICS Resource Definition Guide* for details about specific attributes.

Note: You can assign default values for all attributes of a JOURNALMODEL 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.

JOURNALMODEL(*data-value***)**
specifies the 8-character name of the JOURNALMODEL 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 syntax 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 341 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:

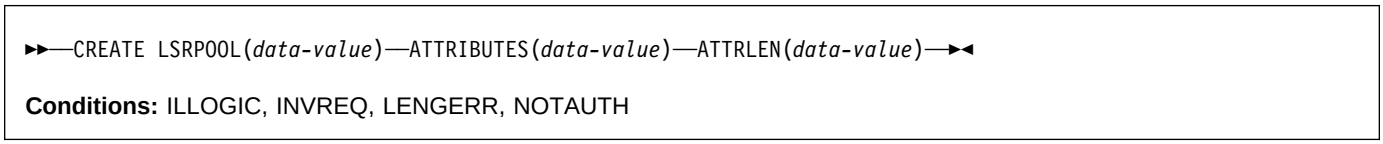
1 The length you have specified in ATTRLEN is negative.

NOTAUTH
RESP2 values:

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

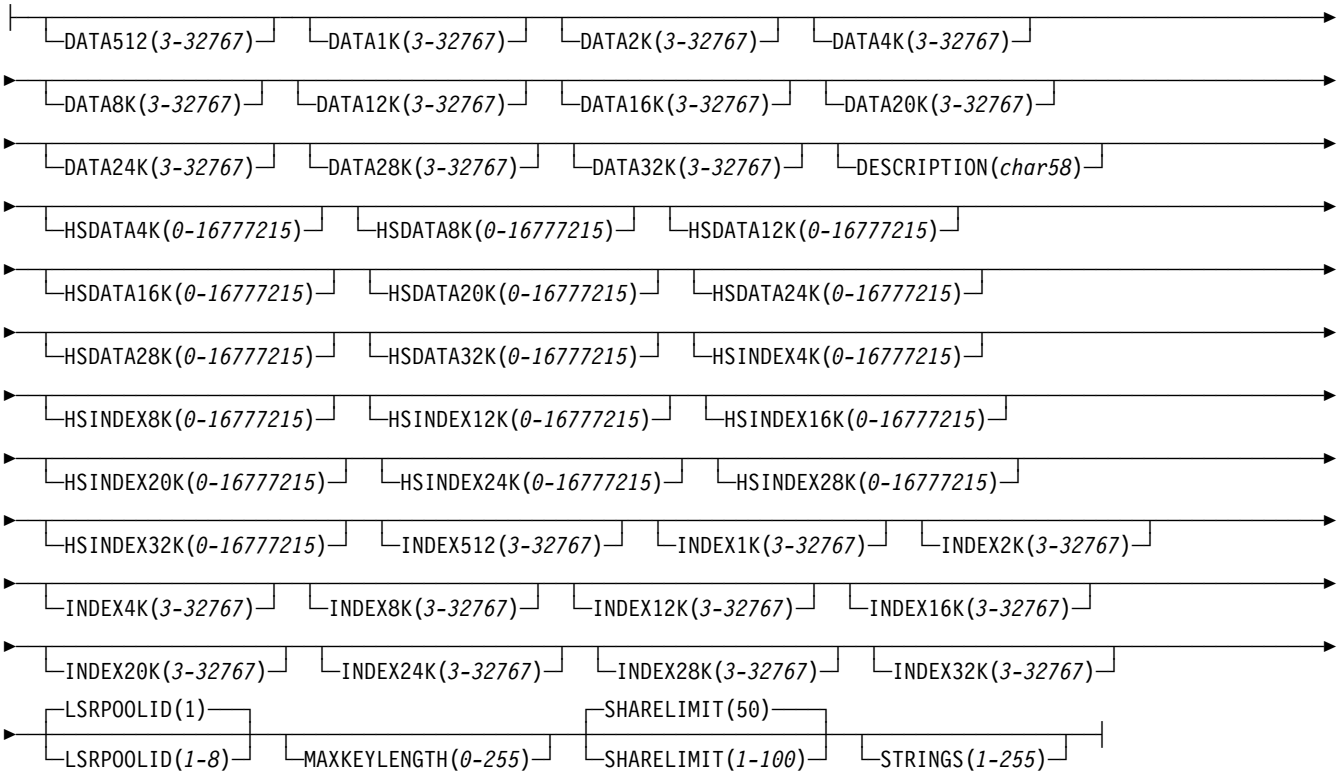
CREATE LSRPOOL

Define an LSR pool in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

LSRPOOL attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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.)

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 22 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 23 for general rules for specifying attributes, and the LSRPOOL chapter in the *CICS 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 syntax 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 341 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:

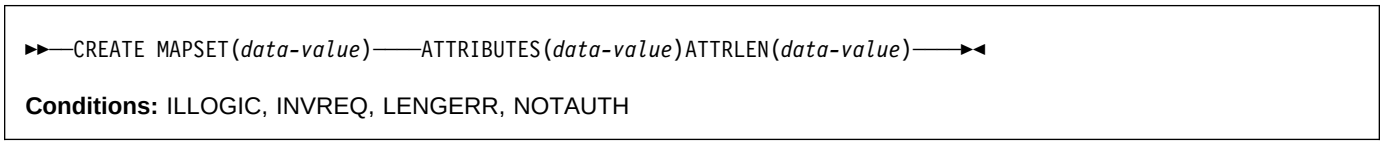
1 The length you have specified in ATTRLEN is negative.

NOTAUTH
RESP2 values:

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

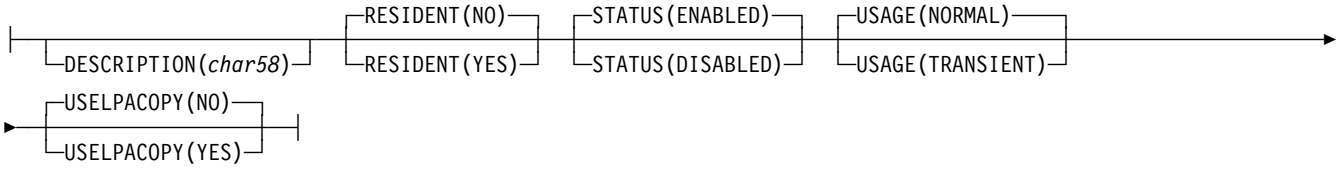
CREATE MAPSET

Define a map set in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

MAPSET attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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 with 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 committed if the CREATE executes successfully, and rolled back if not. See “Creating resource definitions” on page 22 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 23 for general rules for specifying attributes, and the MAPSET chapter in the *CICS 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 syntax 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 341 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

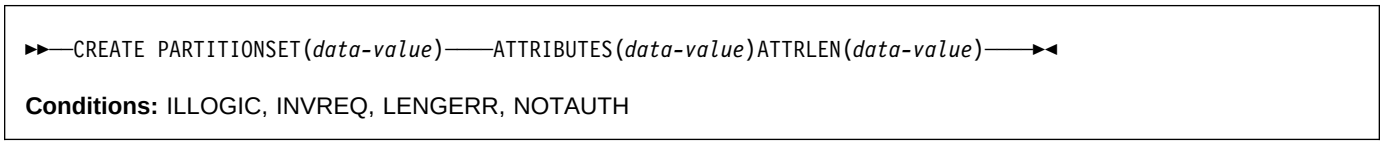
RESP2 values:

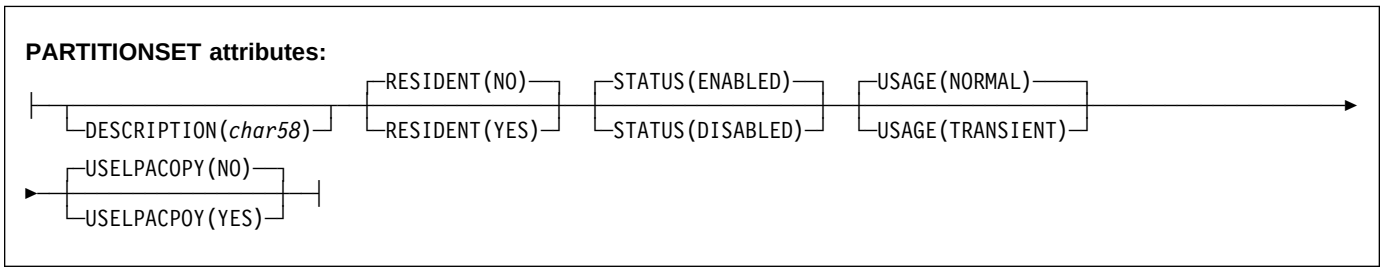
100 The user associated with the issuing task is not authorized to use this command.

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

CREATE PARTITIONSET

Define a partition set in the local CICS region.





Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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 with 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 committed if the CREATE executes successfully, and rolled back if not. See “Creating resource definitions” on page 22 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 23 for general rules for specifying attributes, and the PARTITIONSET chapter in the *CICS 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 syntax 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 341 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

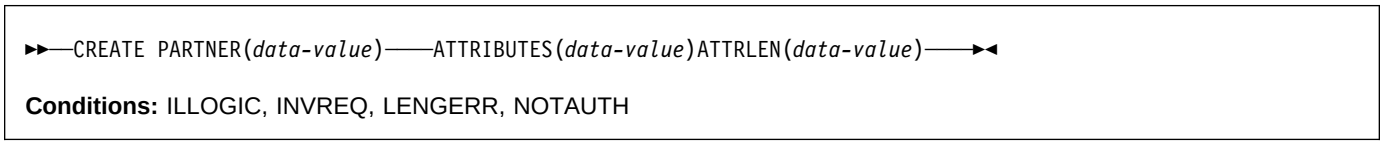
RESP2 values:

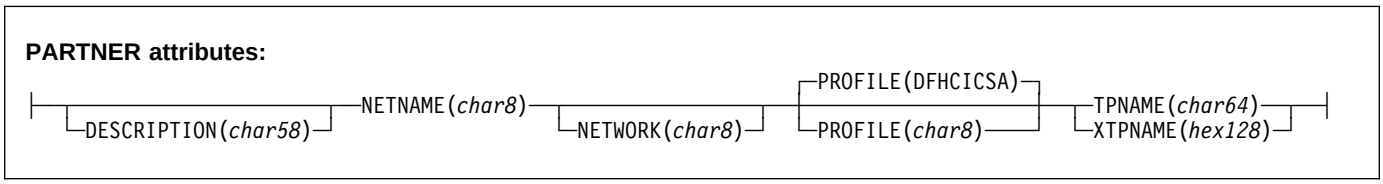
100 The user associated with the issuing task is not authorized to use this command.

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

CREATE PARTNER

Define a PARTNER in the local CICS region.





Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE PARTNER command builds a PARTNER definition, without reference to data stored on the CSD file. If there is already a partner with 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 22 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 23 for general rules for specifying attributes, and the PARTNER chapter in the *CICS 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.

PARTNER(*data-value***)**
specifies the 8-character name of the PARTNER 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 syntax 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 341 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:

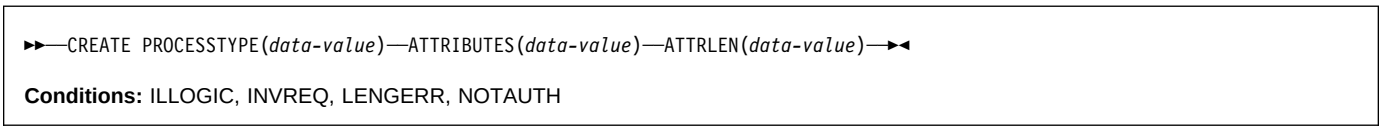
1 The length you have specified in ATTRLEN is negative.

NOTAUTH
RESP2 values:

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

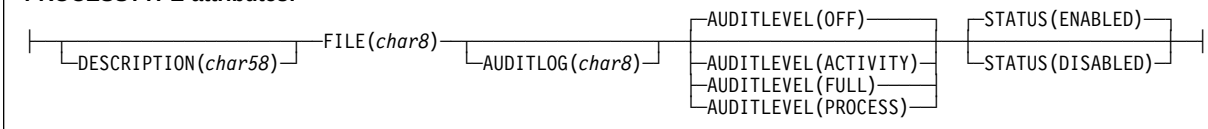
CREATE PROCESSTYPE

Define a PROCESSTYPE in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

PROCESSTYPE attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE PROCESSTYPE command builds a CICS business transaction services (BTS) PROCESSTYPE definition, without reference to data stored on the CSD file. If there is already a process-type with 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 PROCESSTYPE 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the PROCESSTYPE being added. The list of attributes must be coded as a single character string using the syntax shown in **PROCESSTYPE attributes**. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the PROCESSTYPE chapter in the *CICS 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.

PROCESSTYPE(*data-value***)**
specifies the 1- to 8-character name of the PROCESSTYPE definition to be added to the CICS region. The acceptable characters are A-Z a-z 0-9 $ @ # . / -_ % & ? ! : | " = ¬ , ; < >. Leading and embedded blank characters are not permitted. If the name supplied is less than eight characters, it is padded with trailing blanks up to eight characters.

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 syntax 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 341 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.

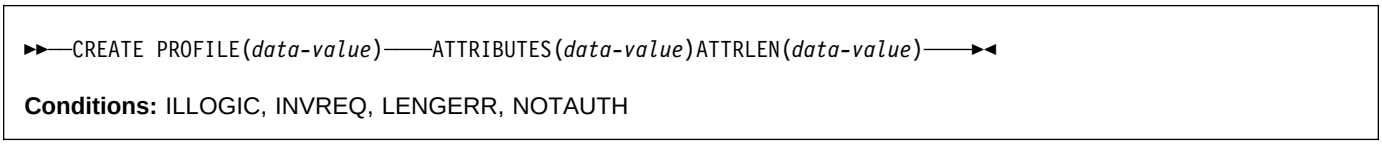
NOTAUTH
RESP2 values:

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

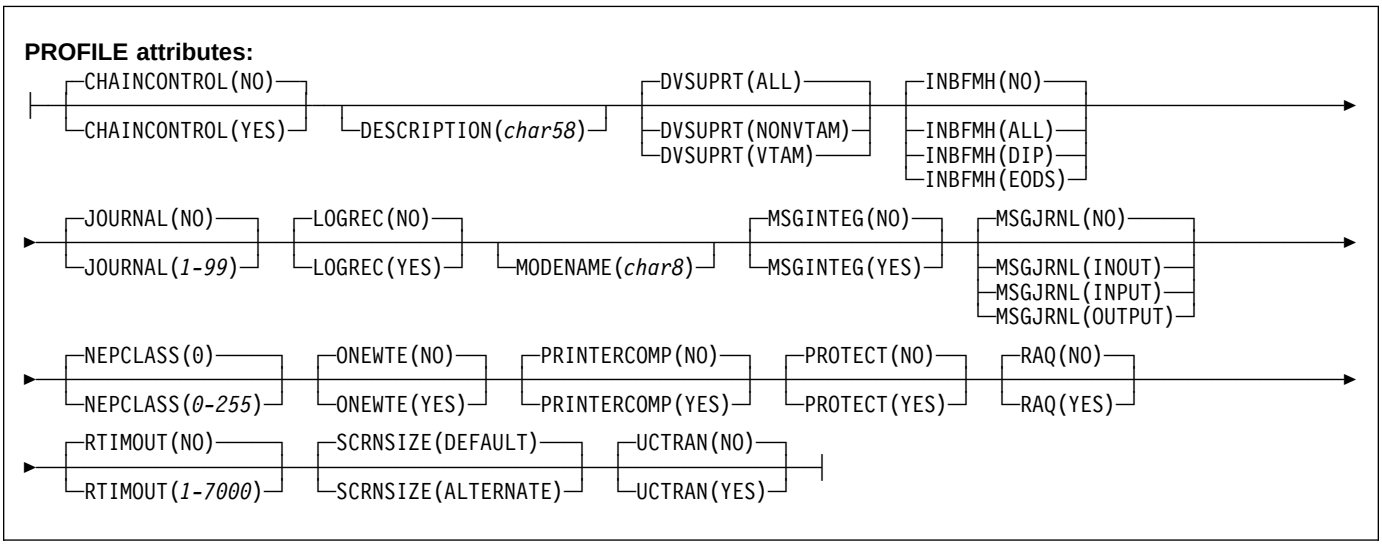
| **101** The user associated with the issuing task is not
| authorized to create a PROCESSTYPE definition
| with this name.
| **102** The caller does not have surrogate authority to
| install the resource with the particular userid.

CREATE PROFILE

Define a transaction PROFILE in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE PROFILE command builds a PROFILE definition, without reference to data stored on the CSD file. If there is already a profile with 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 22 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 23 for general rules for specifying attributes, and the PROFILE chapter in the *CICS 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.

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 syntax 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 341 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:

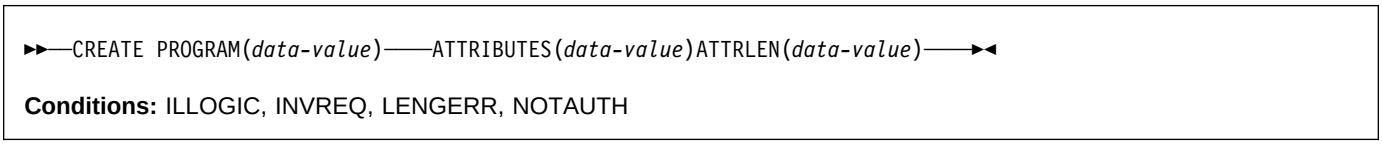
1 The length you have specified in ATTRLEN is negative.

NOTAUTH
RESP2 values:

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

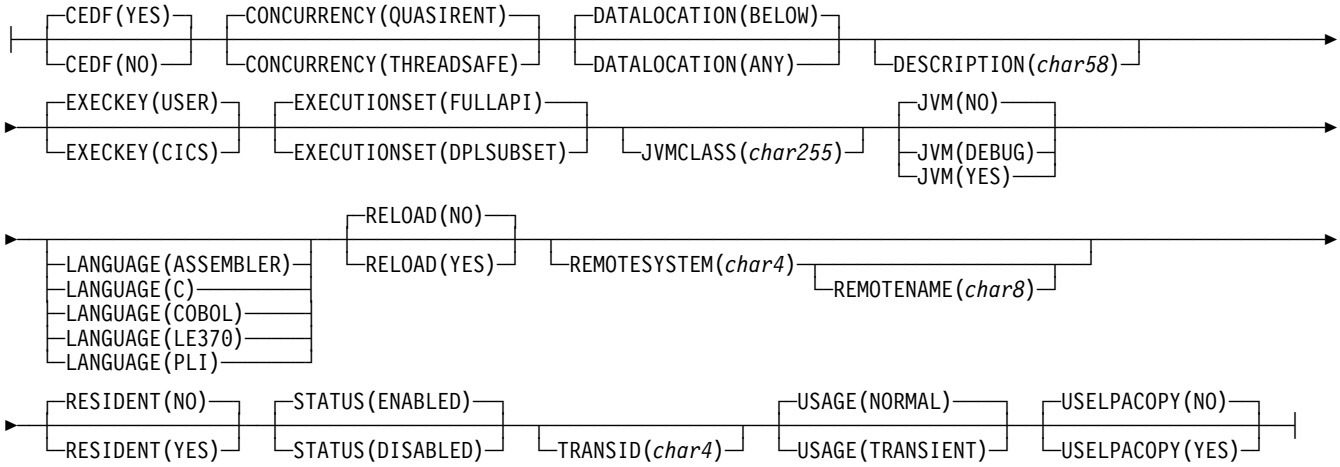
CREATE PROGRAM

Define a PROGRAM in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

PROGRAM attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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 with 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 committed if the CREATE executes successfully, and rolled back if not. See “Creating resource definitions” on page 22 for other general rules governing CREATE commands.

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 23 for general rules for specifying attributes, and the PROGRAM chapter in the *CICS 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.

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 syntax 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 341 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

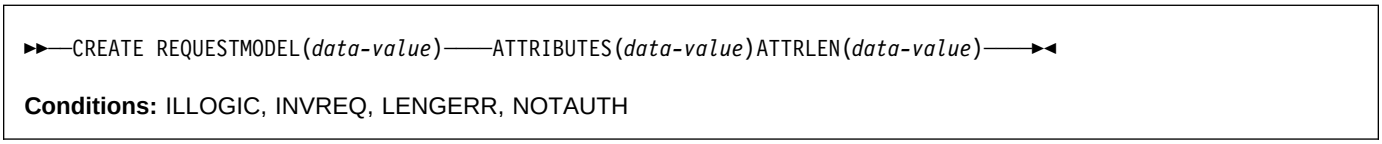
RESP2 values:

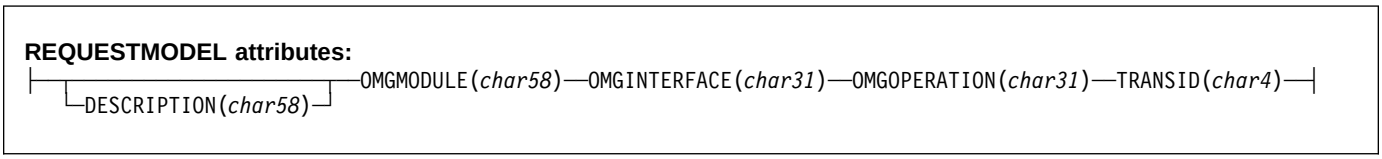
100 The user associated with the issuing task is not authorized to use this command.

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

CREATE REQUESTMODEL

Define a request model in the local CICS region.





Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE REQUESTMODEL command builds a REQUESTMODEL definition, without reference to data stored on the CSD file. If there is already a request model with the name you specify in the local CICS region, the new definition replaces the old one; it not, the new definition is added.

A syncpoint is implicit in CREATE REQUESTMODEL 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the REQUESTMODEL being added. The list of attributes must be coded as a single character string using the syntax shown in **REQUESTMODEL attributes**. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the REQUESTMODEL chapter in the *CICS 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 can be from 0 to 32767.

REQUESTMODEL(*data-value***)**
specifies the 8-character name of the REQUESTMODEL 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 syntax 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 341 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:

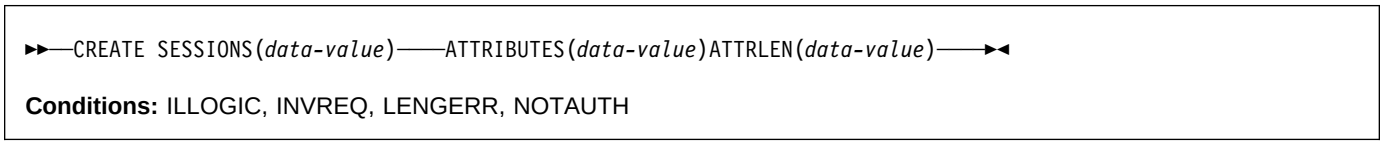
1 The length you have specified in ATTRLEN is negative.

NOTAUTH
RESP2 values:

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

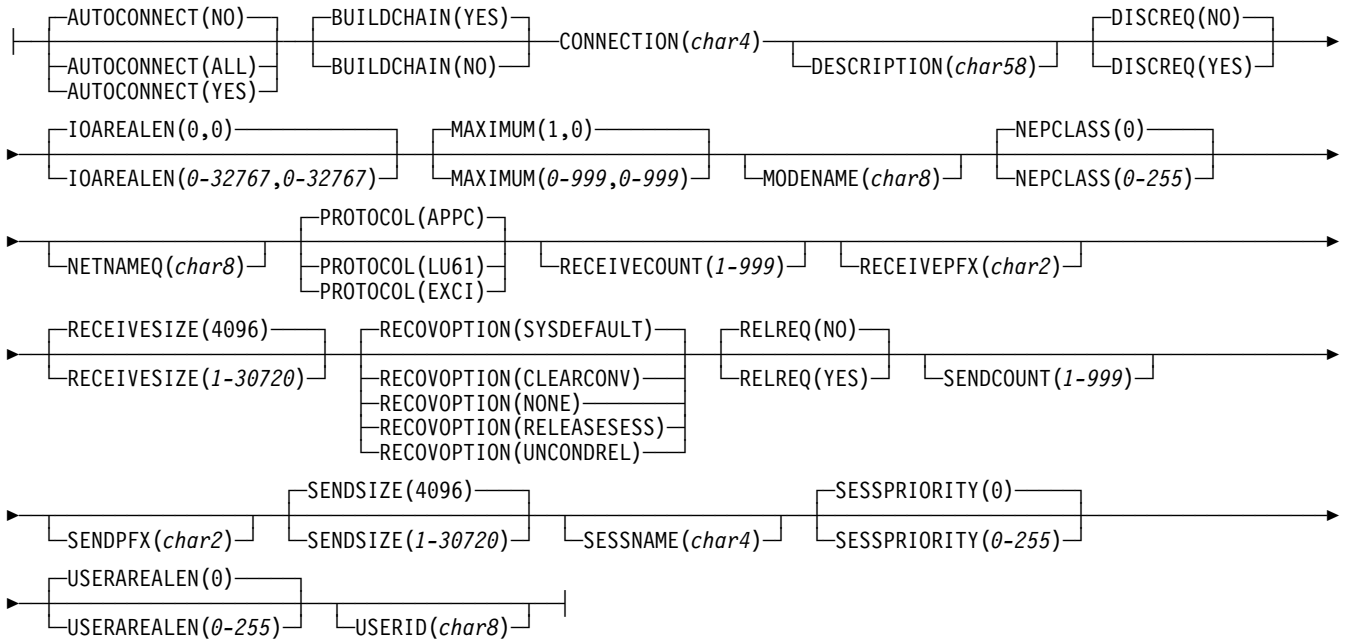
CREATE SESSIONS

Add a session group to the CONNECTION definition being created.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

SESSIONS attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE SESSIONS command defines a group of sessions within a CONNECTION definition that is 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 “CREATE CONNECTION” on page 37 for rules about the order of the commands that build a connection, and “Creating resource definitions” on page 22 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 23 for general rules for specifying attributes, and the SESSIONS chapter in the *CICS 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 syntax 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 341 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

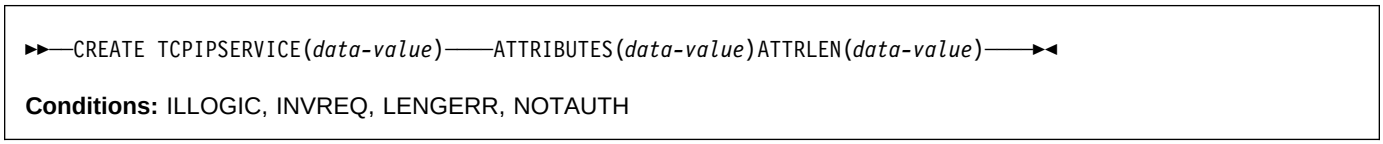
RESP2 values:

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.

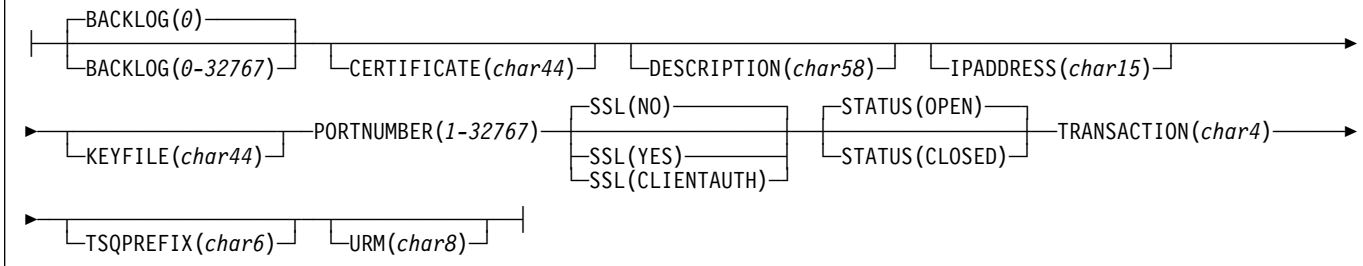
CREATE TCPIPSERVICE

Define a TCP/IP service in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

TCPIPSERVICE attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

CREATE TCPIPSERVICE builds a service definition, without reference to data stored on the CSD file. If there is already a TCP/IP service 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 TCPIPSERVICE 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the TCPIPSERVICE being added. The list of attributes must be coded as a single character string. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the TDQUEUE chapter in the *CICS 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.

TCPIPSERVICE(*data-value***)**
specifies the 16-character name of the TCPIPSERVICE 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 syntax 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 341 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:

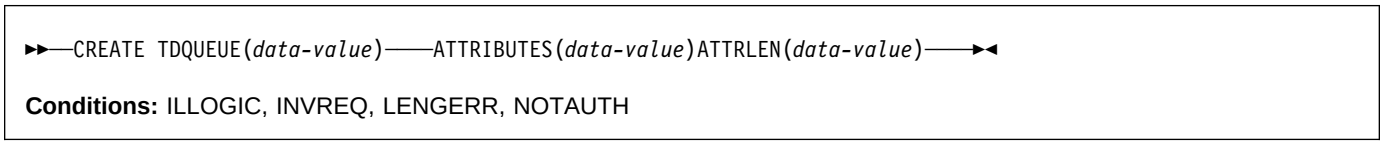
1 The length you have specified in ATTRLEN is negative.

| **NOTAUTH**
| RESP2 values:

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

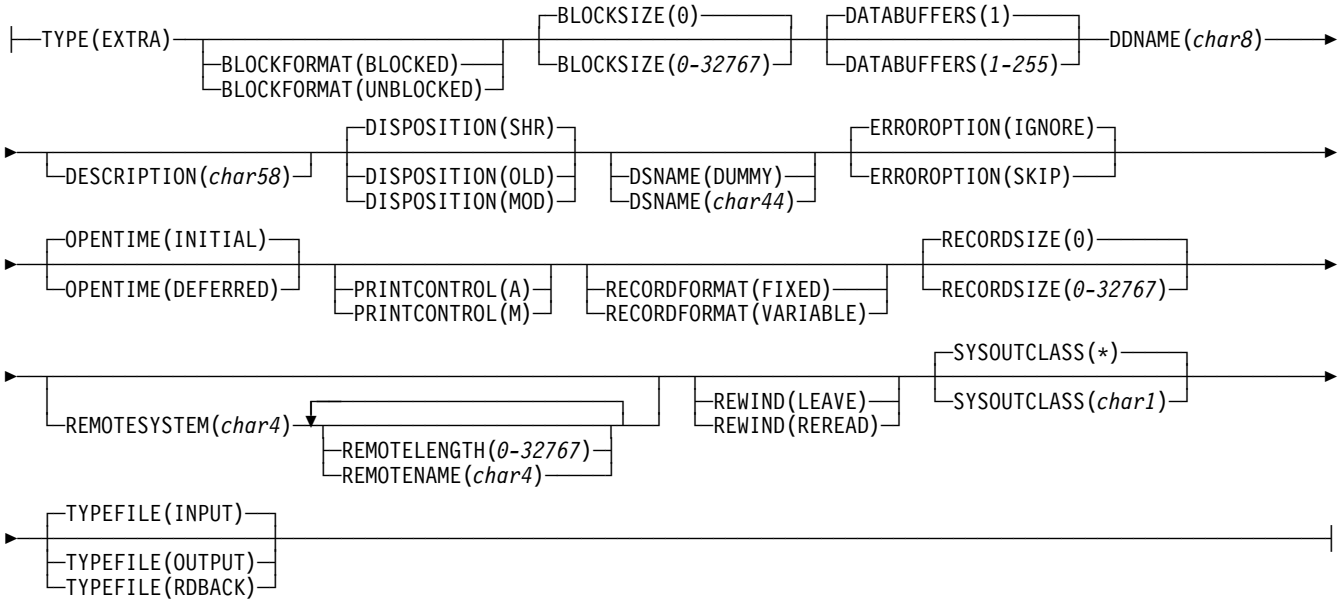
CREATE TDQUEUE

Define a transient data queue in the local CICS region.

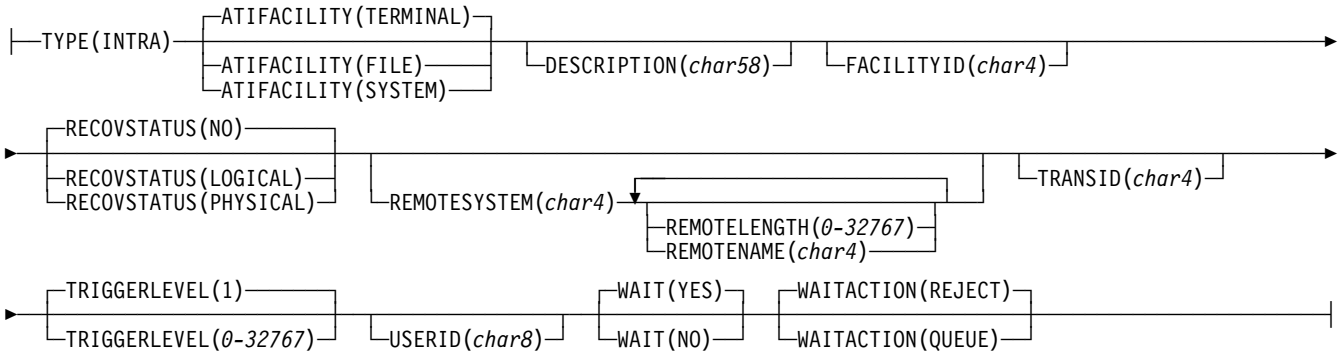


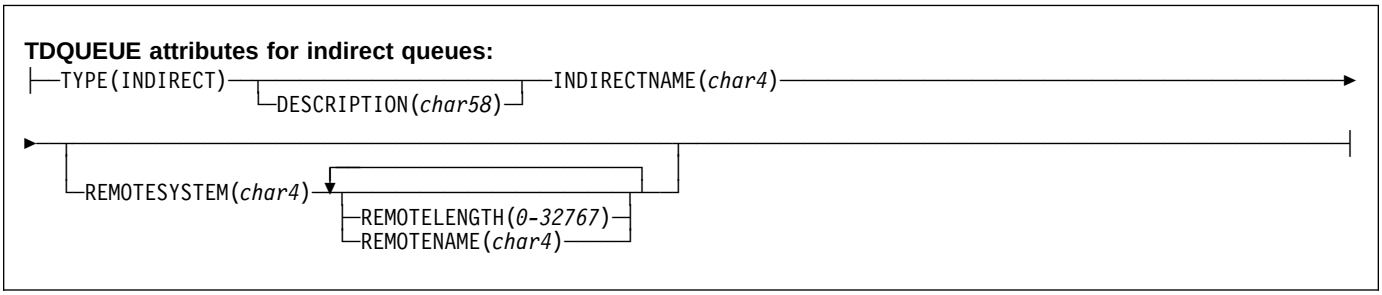
Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

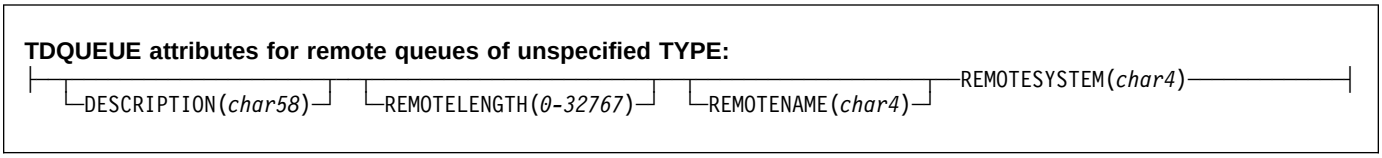
TDQUEUE attributes for extra-partition queues:



TDQUEUE attributes for intra-partition queues:







Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE TDQUEUE command builds a TDQUEUE definition, without reference to data stored on the CSD file. If there is already a transient data queue with 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 TDQUEUE 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the queue being added. The list of attributes must be coded as a single character string and must include the TYPE option unless the queue is remote. The remaining attributes depend on the queue type; use the syntax shown in the figure (**extra-partition, intra-partition,** or **indirect**) that corresponds to your TYPE value. If the queue is remote, you still can specify TYPE and use the appropriate syntax, but you can also use the briefer form labelled **remote queues of unspecified TYPE**. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the TDQUEUE chapter in the *CICS 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.

TDQUEUE(*data-value***)**
specifies the 4-character name of the TDQUEUE 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 syntax 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 341 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:

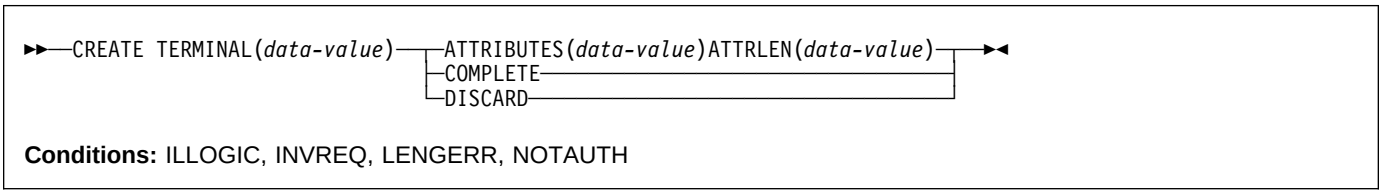
100 The user associated with the issuing task is not authorized to use this command.

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

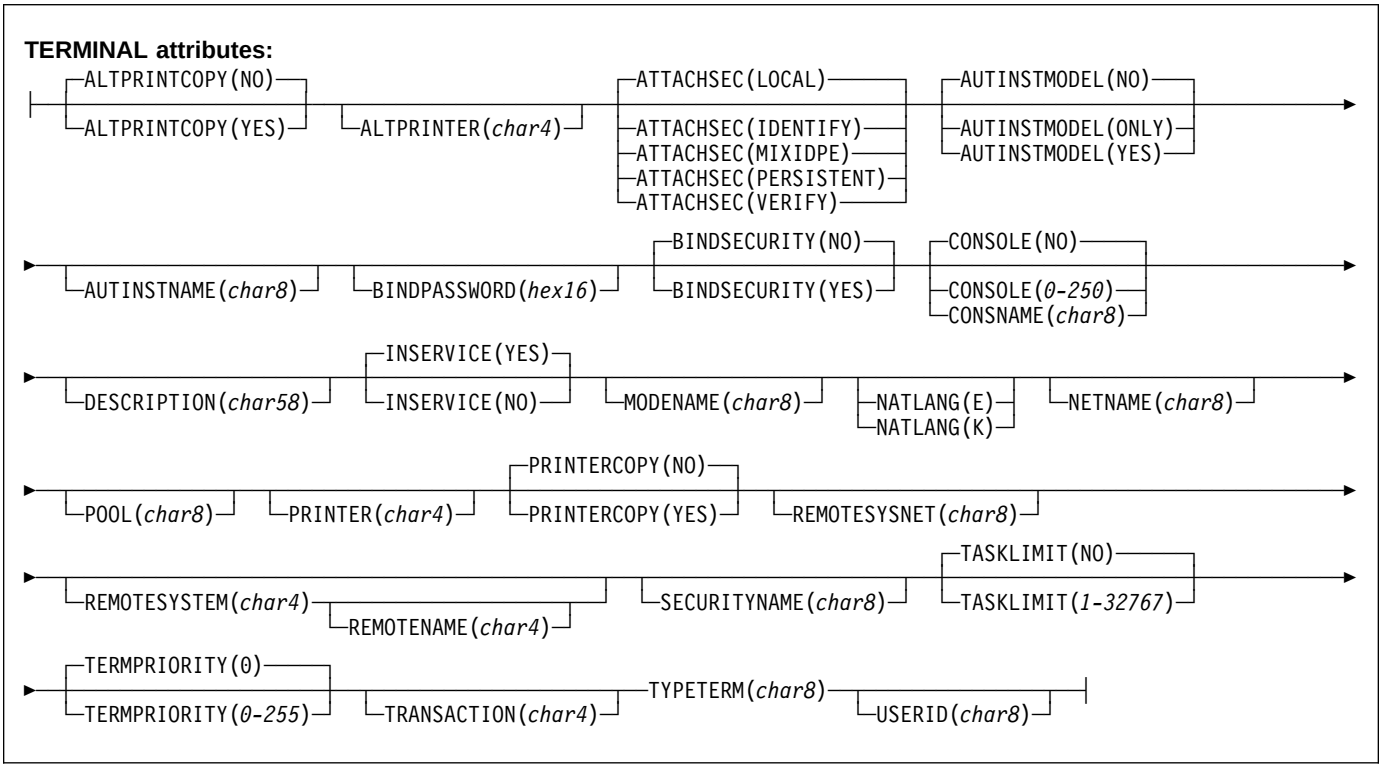
102 The user of the transaction issuing the CREATE TDQUEUE command is not an authorized surrogate of the user specified in USERID.

CREATE TERMINAL

Define a TERMINAL in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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 with 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. 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 discards 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 22 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 23 for general rules for specifying attributes, and the TERMINAL chapter in the *CICS 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.

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

TERMINAL(*data-value***)**
specifies the 4-character name of the TERMINAL definition to be 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 syntax 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 341 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.

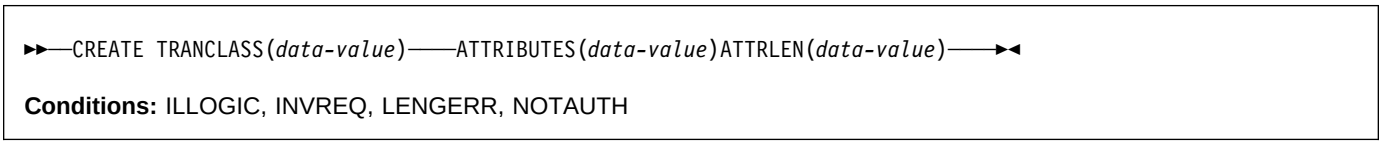
NOTAUTH
RESP2 values:

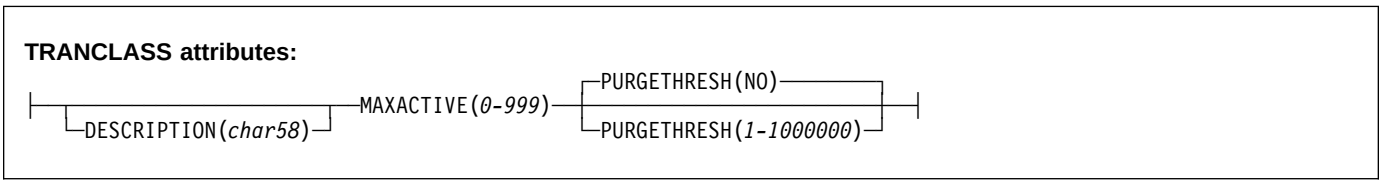
100 The user associated with the issuing task is not authorized to use this command.

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

Define a transaction class in the local CICS region.





Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE TRANCLASS command builds a TRANCLASS definition, without reference to data stored on the CSD file. If there is already a transaction class with 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 22 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 23 for general rules for specifying attributes, and the TRANCLASS chapter in the *CICS 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.

TRANCLASS(*data-value***)**
specifies the 8-character name of the TRANCLASS 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 syntax 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 341 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:

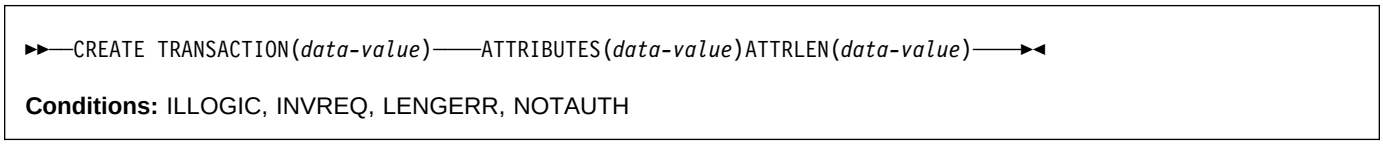
1 The length you have specified in ATTRLEN is negative.

NOTAUTH
RESP2 values:

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

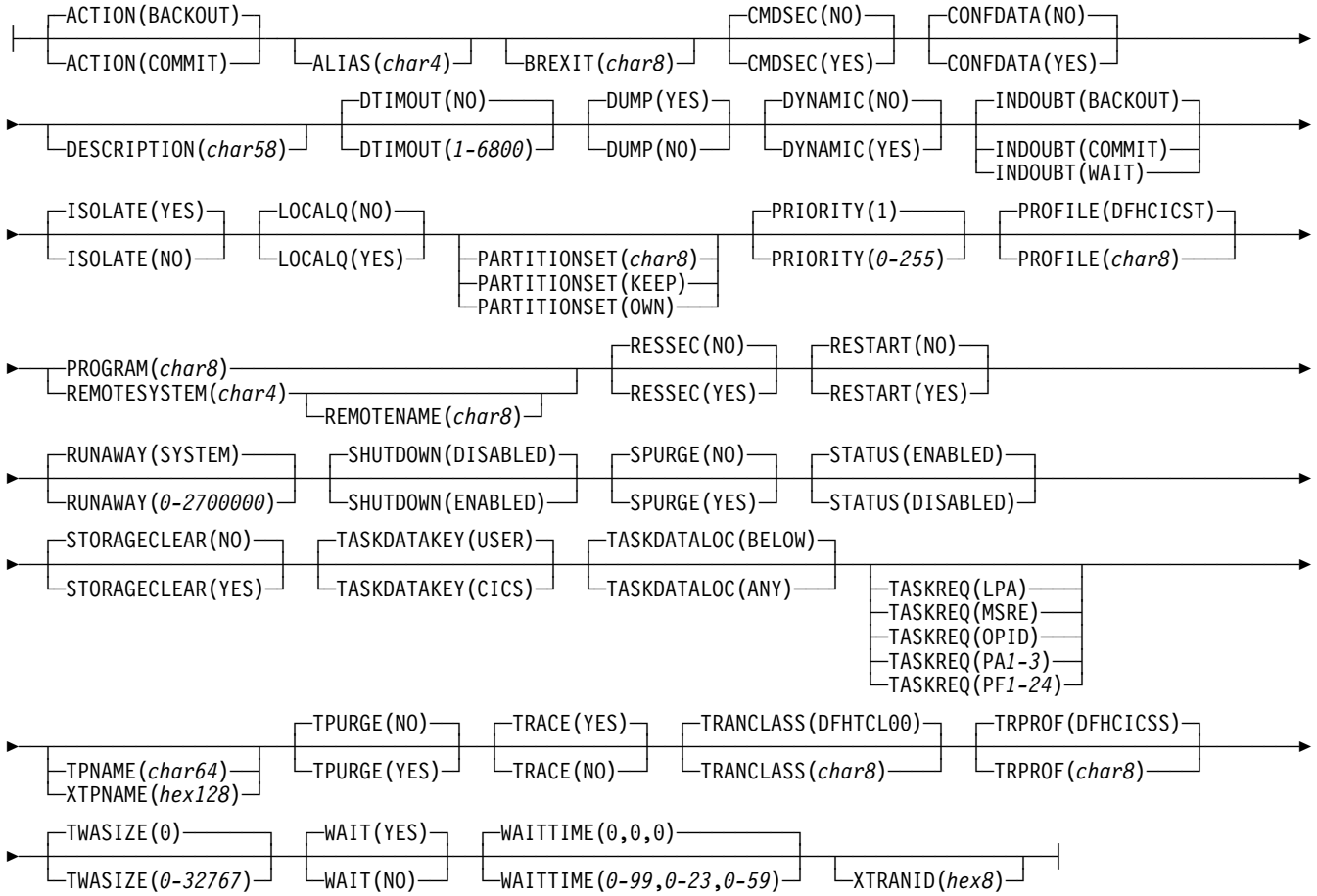
CREATE TRANSACTION

Define a TRANSACTION in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

TRANSACTION attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

The CREATE TRANSACTION command builds a TRANSACTION definition, without reference to data stored on the CSD file. If there is no transaction with 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.

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 22 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 23 for general rules for specifying attributes, and the TRANSACTION chapter in the *CICS 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 syntax 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 341 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.

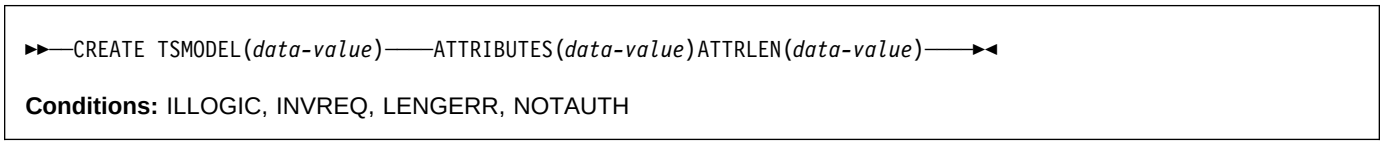
NOTAUTH
RESP2 values:

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.

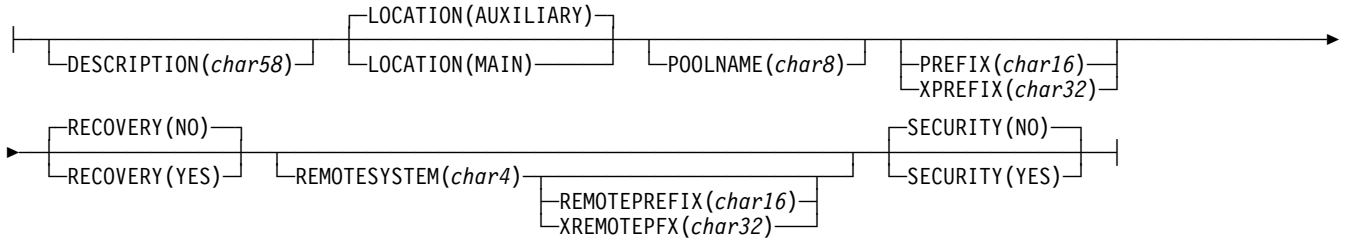
CREATE TSMODEL

Define a temporary Storage Table in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

TSMODEL attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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

No two TS models may have the same prefix. An attempt to add or replace a model which would result in there being two models with the same prefix will therefore fail.

A syncpoint is implicit in CREATE TSMODEL 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 22 for other general rules governing CREATE commands.

Options

ATTRIBUTES(*data-value***)**
specifies the attributes of the TSMODEL being added. The list of attributes must be coded as a single character string using the syntax shown in **TSMODEL attributes**. See “ATTRIBUTES option” on page 23 for general rules for specifying attributes, and the TSMODEL chapter in the *CICS 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.

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

Conditions

INVREQ
RESP2 values:

n There is a syntax 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 341 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:

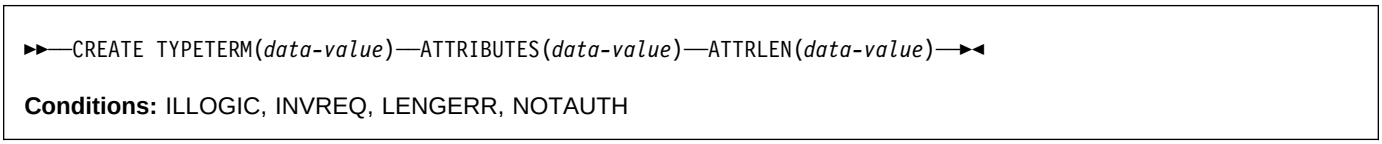
1 The length you have specified in ATTRLEN is negative.

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 TSMODEL definition with this name.

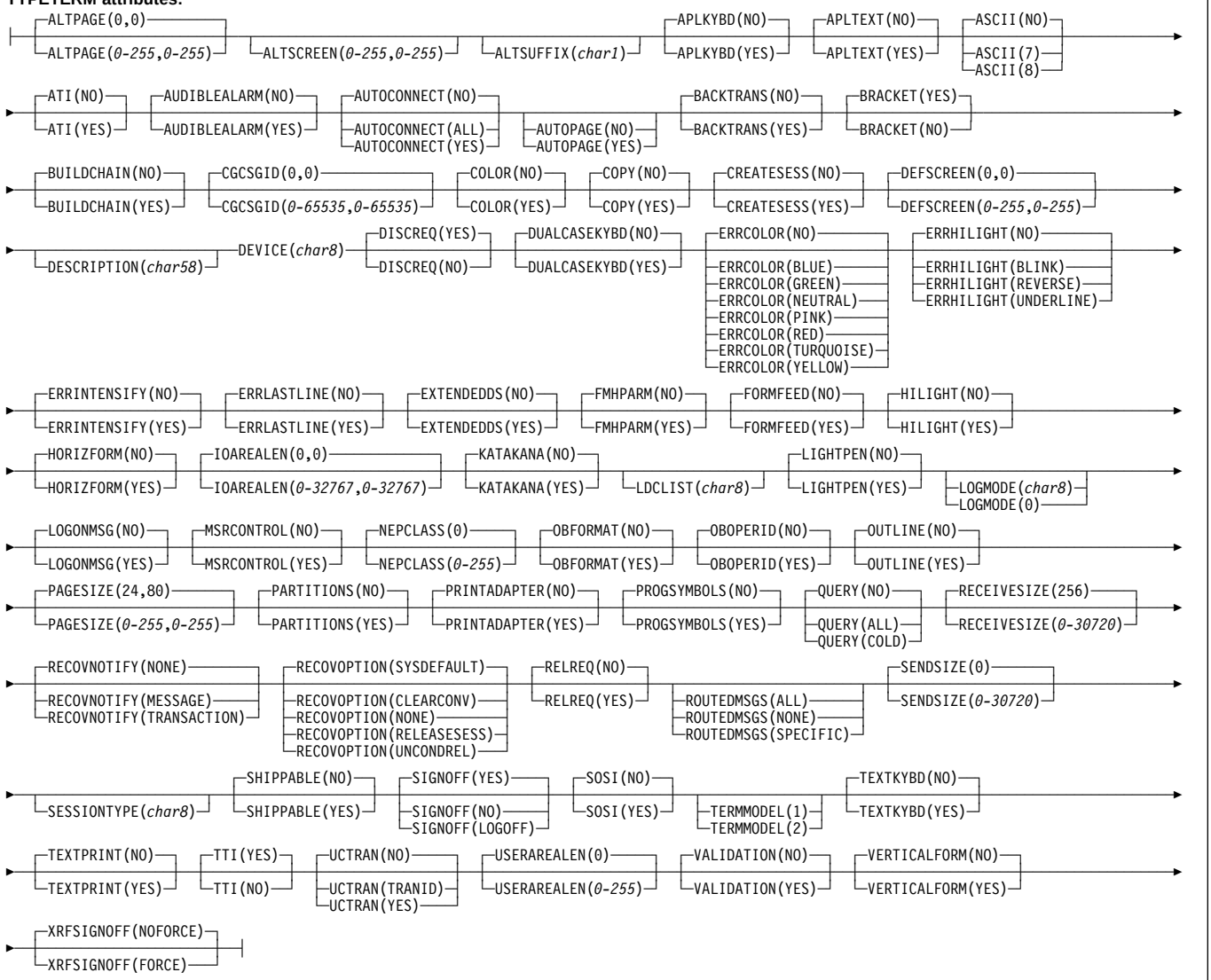
CREATE TYPETERM

Define a terminal type in the local CICS region.



Conditions: ILLOGIC, INVREQ, LENGERR, NOTAUTH

TYPETERM attributes:



Note to COBOL programmers:

In the syntax above, you must use
--- ATTRIBUTES (data-area)--- instead of
--- ATTRIBUTES (data-value)---

Description

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 with 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 22 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 23 for general rules for specifying attributes, and the TYPETERM chapter in the *CICS 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 syntax 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 341 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:

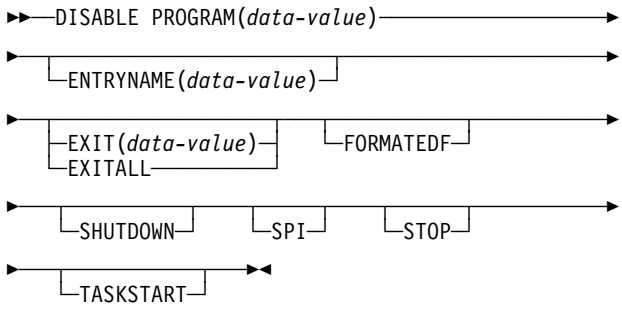
1 The length you have specified in ATTRLEN is negative.

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.



Conditions: INVEXITREQ, NOTAUTH

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 Customization Guide*; you should also see the general discussion of commands that modify exits in “Exit-related commands” on page 25.

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 program 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 eight characters long; for programming information, including a list of exit points, see the *CICS 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.

SPI (task-related user exits only)
specifies that the task-related user exit is no longer to be invoked if an INQUIRE EXITPROGRAM command specifies the CONNECTST or QUALIFIER option, or both.

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 is still 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 is deleted.

DISCARD AUTINSTMODEL

Remove a terminal autoinstall model definition.

►►—DISCARD AUTINSTMODEL(*data-value*)—►◄

Conditions: INVREQ, MODELIDERR, NOTAUTH

Description

The DISCARD AUTINSTMODEL command makes a TERMINAL definition in the local CICS system ineligible for use as a model for automatic installation of terminals. The TERMINAL definition is not discarded or otherwise modified; it is only removed from the list of autoinstall models available. (Use the DISCARD TERMINAL command if you want to remove the definition of the terminal.)

See “Discarding resource definitions” on page 24 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, or the name of the terminal if AUTINSTNAME was not specified.

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 CONNECTION

Remove a CONNECTION definition.

►►—DISCARD CONNECTION(*data-value*)—►◄

Conditions: INVREQ, NOTAUTH, SYSIDERR

Description

The DISCARD CONNECTION command removes a CONNECTION definition from the local CICS system. When a connection is removed, all of the associated sessions also are removed.

For deletion to be successful:

- The connection can have no active sessions if it is remote (that is, has a REMOTESYSTEM value other than the name of the local region), and must be in OUTSERVICE status if it is not remote.
- The interregion communications facility must be closed if the connection is an MRO connection. (You can use the SET IRC CLOSED command to close it.)
- If the connection is an APPC connection and the local CICS system is a member of a VTAM generic resource group, there can be no deferred work pending. Deferred work occurs when a failure causes a unit of work which has used a session on the connection at SYNCLEVEL 2 to be “shunted” (held for later disposition, because recovery action is required before disposition can be completed).

  Other types of connection *can* be discarded, even if there is recovery work outstanding for them. However, it is recommended that you do not discard them if there is. You can use the INQUIRE CONNECTION RECOVSTATUS command to check.
- There can be no indirect CONNECTION definitions pointing to the connection to be discarded.

Note: In unusual circumstances, the discard of an LU6.1 connection can fail, even when it is out-of-service, if some of its sessions are still in-service. If this happens, set the connection status to INSERVICE, then OUTSERVICE, and then reissue the DISCARD command.

CICS completes successful DISCARD CONNECTION processing with an implicit syncpoint on behalf of the issuing task, committing changes to recoverable resources made up to that point in the task. If the discard processing fails, CICS raises the INVREQ exception condition with a RESP2 value of 27, and does a SYNCPOINT ROLLBACK instead, rolling back changes to recoverable resources. For all other exception conditions, however, discard processing is not

attempted and neither SYNCPOINT nor SYNCPOINT ROLLBACK is issued.

See “Discarding resource definitions” on page 24 for general information about DISCARD commands.

Options

CONNECTION(*data-value***)**
specifies the 4-character identifier of the CONNECTION definition to be discarded.

Conditions

INVREQ
RESP2 values:

24 The connection is remote and is in use locally.
25 The connection is local and is not out-of-service.
26 Recovery information is outstanding for the connection which must be resolved before discard is allowed.
27 Discard processing failed.
28 Indirect connections point to the connection.
29 The connection is an MRO connection and IRC is not closed.
38 Discard of this connection is already in progress.
39 The CONNECTION definition is currently in use.
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.

NOTAUTH
RESP2 values:

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

SYSIDERR
RESP2 values:

9 The connection cannot be found.

DISCARD DB2CONN

Remove a DB2CONN definition.

```
►►─DISCARD DB2CONN─►◄
```

Conditions: INVREQ, NOTAUTH, NOTFND

Description

The DISCARD DB2CONN command removes the definition of a DB2CONN from the local CICS system; that is, it revokes the earlier installation of a DB2CONN resource definition.

A DB2CONN can only be discarded when the CICS DB2 interface is not active.

Note: A discard of a DB2CONN also implicitly discards all DB2ENTRYs and DB2TRANs currently installed.

Options

None

Conditions

INVREQ
RESP2 values:

2 The CICS DB2 interface is active.

NOTAUTH
RESP2 values:

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

NOTFND
RESP2 values:

1 A DB2CONN cannot be found.

DISCARD DB2ENTRY

Remove a DB2ENTRY definition.

```
►►—DISCARD DB2ENTRY(data-value)—►◄
```

Conditions: NOTFND, INVREQ, NOTAUTH

Description

The DISCARD DB2ENTRY command removes the definition of a DB2ENTRY from the local CICS system, so that the system no longer has access to the DB2ENTRY; that is, it revokes the earlier installation of a DB2ENTRY resource definition of the same name.

A DB2ENTRY must be disabled for its definition to be discarded.

Options

DB2ENTRY(*data-value***)**
specifies the 8-character name of the DB2ENTRY that is to be removed.

Conditions

NOTFND
RESP2 values:

1 The DB2ENTRY cannot be found.

INVREQ
RESP2 values:

2 The DB2ENTRY is currently in use.
3 The DB2ENTRY is not disabled.

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 DB2TRAN

Remove a DB2TRAN definition.

```
►►—DISCARD DB2TRAN(data-value)—►◄
```

Conditions: NOTFND, NOTAUTH

Description

The DISCARD DB2TRAN command removes the definition of a DB2TRAN from the local CICS system, so that the transaction id specified in the DB2TRAN no longer uses the named DB2ENTRY; that is, it revokes the earlier installation of a DB2TRAN resource definition of the same name.

A DB2TRAN can be discarded at any time.

Options

DB2TRAN(*data-value***)**
specifies the 8-character name of the DB2TRAN that is to be removed.

Conditions

NOTFND
RESP2 values:

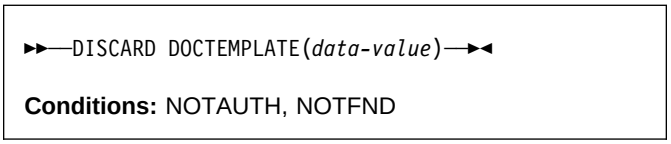
1 The DB2TRAN cannot be found.

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 the DB2ENTRY referenced by this DB2TRAN in the way required by this command.

DISCARD DOCTEMPLATE

Remove a document template.



Conditions: NOTAUTH, NOTFND

Description

The DISCARD DOCTEMPLATE command removes a document template definition from the local CICS system, so that the system no longer has access to the resource (that is, it revokes the earlier installation of an DOCTEMPLATE definition of the same name).

See “Discarding resource definitions” on page 24 for general information about discards.

Options

DOCTEMPLATE(*data-value***)**
specifies the 8-character name of the DOCTEMPLATE definition that you want to remove.

Conditions

NOTAUTH
RESP2 values:

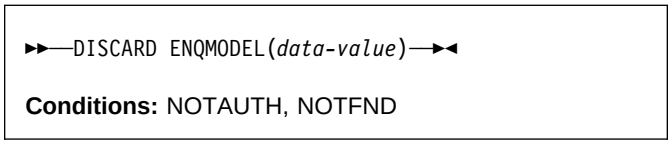
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The specified DOCTEMPLATE is not installed on this system.

DISCARD ENQMODEL

Remove an ENQMODEL resource definition.



Conditions: NOTAUTH, NOTFND

Description

The DISCARD ENQMODEL command removes the definition of an ENQ model from the local CICS system. When discard is issued, the ENQMODEL is put into the WAITING state until there are no enqueues in the local system which match the ENQNAME pattern. It is then removed from the local system, so that the system no longer has access to the ENQMODEL; that is, it revokes the earlier installation of an ENQMODEL resource definition of the same name.

Adding or removing a definition does not affect enqueues already held, only ENQ commands issued after the definition is added or removed are affected.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

ENQMODEL(*data-value***)**
specifies the 8-character identifier of the ENQ model that is to be discarded.

Conditions

NOTAUTH
RESP2 values:

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

NOTFND
RESP2 values:

1 The specified ENQMODEL is not installed on this system.

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 local 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 and disabled for its definition to be discarded. In addition, if the file is recoverable, it cannot be discarded until all retained locks on it are released. A lock is retained when a failure causes a unit of work which has modified the file to be “shunted” (held for later disposition, because recovery action is required before disposition can be completed).

See “Discarding resource definitions” on page 24 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:

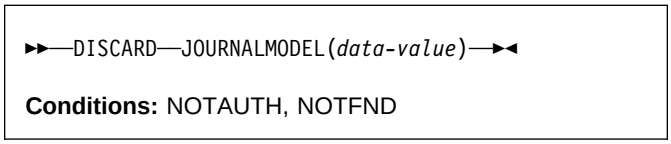
2 The file is not closed.
3 The file is not disabled.
25 The FILE definition is currently in use.
26 The file cannot be discarded because its name begins with DFH.
43 The file cannot be discarded because it has deferred work outstanding, for which there are retained locks.

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 JOURNALMODEL

Remove a journal model definition.



Conditions: NOTAUTH, NOTFND

Description

The DISCARD JOURNALMODEL command makes a JOURNALMODEL definition ineligible as a model for defining journals in local CICS system. The JOURNALMODEL definition itself is not discarded or otherwise modified, nor is there any effect on existing journals defined using the model. These journals continue to use their existing definitions unless they are discarded using a DISCARD JOURNALNAME command.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

JOURNALMODEL(*data-value***)**
specifies the 8-character name of the journal model that you want to remove.

Conditions

NOTAUTH
RESP2 values:

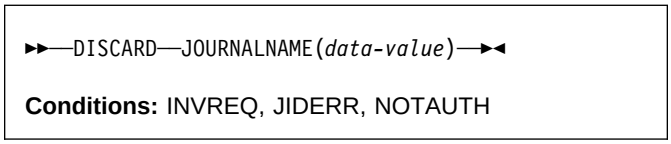
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The journal model name was not found.

DISCARD JOURNALNAME

Remove a journal name from the journal names table.



Conditions: INVREQ, JIDERR, NOTAUTH

Description

The DISCARD JOURNALNAME command removes a journal definition from the local CICS system, so that the next time the journal definition is used, it is recreated based on the current set of JOURNALMODEL definitions. Thus you can use it in conjunction with DISCARD and CREATE JOURNALMODEL commands to change the definition of a particular journal.

The command takes effect immediately for user journals, including the “log of logs” journal, and for terminal control autojournals. On the next reference to the journal following the DISCARD, a new journal definition is created using attributes from the JOURNALMODEL definition that matches best at that time.

For forward recovery and auto-journaling journals, however, the journal definition is used only when one of the files using the journal is opened. Hence the command has no effect on forward-recovery logging or auto-journaling operations for VSAM files that are open and using the journal at the time of the DISCARD. They continue to use the log stream referenced by the existing journal until the files are closed, and are not affected by the DISCARD unless the file is subsequently reopened. In addition, if the logstream identifier is present in the VSAM catalog definition for a file, as it must be for an RLS file and may be for others, the catalog value overrides the JOURNALMODEL value.

Neither component of the CICS system log, DFHLOG or DFHSHUNT, is eligible for discard.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

JOURNALNAME(*data-value***)**
specifies the 8-character name of the journal that you want to remove.

Note: To discard a journal defined with a numeric identifier specify the journal name as DFHJ*nn*, where *nn* is the two-digit journal number, in the range 01–99. (DFHJ01 is a user journal in CICS Transaction Server for OS/390, not the system log.)

Conditions

INVREQ
RESP2 values:

3 The journal specified cannot be discarded.

JIDERR
RESP2 values:

1 The journal cannot 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.

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 local 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 24 for general information about discards.

Options

PARTNER(*data-value***)**
specifies the 8-character name of the partner that is to be removed.

Partners whose names begin with the letters DFH are assumed to be CICS-defined partners and cannot be discarded.

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 PROCESSTYPE

Remove a PROCESSTYPE definition.

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

Conditions: INVREQ, NOTAUTH, PROCESSERR

Description

The DISCARD PROCESSTYPE command removes a CICS business transaction services (BTS) PROCESSTYPE definition from the local CICS region.

Notes:

1. Only disabled process-types can be discarded.
2. If you are using BTS in a single CICS region, you can use the DISCARD PROCESSTYPE command to remove process-types. However, if you are using BTS in a sysplex, it is strongly recommended that you use CPSM to remove them. If you don’t use CPSM, problems could arise if Scheduler Services routes to this region work that requires a discarded definition.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

PROCESSTYPE(*data-value***)**
specifies the 8-character name of the PROCESSTYPE that you want to remove.

Conditions

INVREQ
RESP2 values:

2 The process-type named in the PROCESSTYPE option is not disabled.

NOTAUTH
RESP2 values:

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

PROCESSERR
RESP2 values:

1 The process-type named in the PROCESSTYPE option is not defined in the process-type table (PTT).

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 local 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 24 for general information about discards.

Options

PROFILE(*data-value***)**
specifies the 8-character name of the profile that is to be removed.

Profiles whose names begin with the letters DFH are assumed to be CICS-supplied profiles 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 the definition of a program, map set, or partition set.

```
►►─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 local 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 24 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:

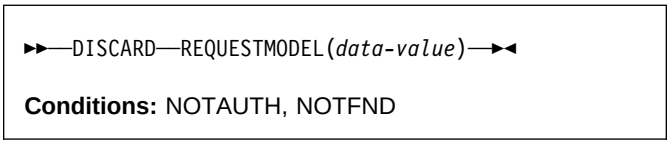
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

PGMIDERR
RESP2 values:

7 The resource definition cannot be found.

DISCARD REQUESTMODEL

Remove a request model definition.



Conditions: NOTAUTH, NOTFND

Description

The DISCARD REQUESTMODEL command makes a REQUESTMODEL definition ineligible as a model for defining requests in the local CICS system. The REQUESTMODEL definition itself is not deleted or otherwise modified.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

REQUESTMODEL(*data-value***)**
specifies the 8-character name of the request model that you want to remove.

Conditions

NOTAUTH
RESP2 values:

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

NOTFND
RESP2 values:

1 The request model name was not found.

DISCARD TCPIPSERVICE

Remove a TCPIPSERVICE definition.

Conditions: INVREQ, NOTAUTH, NOTFND

Description

The DISCARD TCPIPSERVICE command removes a TCPIPSERVICE definition from the local CICS system.

You cannot discard a TCPIPSERVICE unless it is in CLOSED status, showing that is not in use.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

TCPIPSERVICE(*data-value***)**
specifies the 8-character name of the TCPIPSERVICE that you want to remove.

Conditions

INVREQ
RESP2 values:

9 The TCPIPservice is still open.

NOTAUTH
RESP2 values:

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

NOTFND
RESP2 values:

3 The TCPIPSERVICE name was not found.

DISCARD TDQUEUE

Remove a transient data queue definition.

```
►►─DISCARD TDQUEUE(data-value)─►◄
```

Conditions: INVREQ, NOTAUTH, QIDERR

Description

The DISCARD TDQUEUE command removes the definition of a transient data queue from the local CICS system.

A queue must be disabled before it can be discarded, and an extrapartition queue must be closed as well. See “SET TDQUEUE” on page 296 for rules governing disabling of queues. Queues required by CICS (those whose names begin with the letter C) cannot be discarded.

When an intrapartition queue is discarded, an implicit DELETEQ command is executed to empty the queue and release space in the data set associated with it. If the queue is defined as logically recoverable, an implicit SYNCPOINT command follows the DELETEQ. The SYNCPOINT commits all changes to recoverable resources made up to that point in the task that issued the DISCARD TDQUEUE command. However, deletion and syncpoint take place only if the command completes successfully, without raising any exception conditions.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

TDQUEUE(*data-value***)**
specifies the 4-character name of the transient data queue that is to be removed.

Conditions

INVREQ
RESP2 values:

11 The queue name begins with the letter C.
18 The queue is not closed.
30 The queue is in “disable pending” status (that is, the disabling process is not completed).
31 The queue is not disabled.
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.

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 queue cannot be found.

DISCARD TERMINAL

Remove a TERMINAL definition.

```
►►─DISCARD TERMINAL(data-value)─►◄
```

Conditions: INVREQ, NOTAUTH, TERMIDERR

Description

The DISCARD TERMINAL command removes the definition of a terminal from the local CICS system, so that the system no longer has access to the terminal; that is, it deletes a TERMINAL resource definition of the same name which was installed explicitly, installed automatically, or shipped by another CICS which routed a transaction to the local CICS.

To be eligible for discard, a terminal defined as local must be either a VTAM terminal or a console, it must be in out-of-service status, and it cannot be the CICS-defined error console CERR. A remote terminal cannot be in use by the local system (that is, it cannot be the principal facility of a task there). Sessions on a connection cannot be discarded with a DISCARD TERMINAL command, even if they were installed via a TERMINAL resource definition. You must use DISCARD CONNECTION instead.

CICS completes successful DISCARD TERMINAL processing with an implicit syncpoint on behalf of the issuing task, committing changes to recoverable resources made up to that point in the task. If the discard processing fails, CICS raises the INVREQ exception condition with a RESP2 value of 43, and does a SYNCPOINT ROLLBACK instead, rolling back changes to recoverable resources. In all other exception situations, however, discard processing is not attempted and neither SYNCPOINT nor SYNCPOINT ROLLBACK is issued.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

TERMINAL(*data-value***)**
specifies the 4-character name of the terminal whose definition is to be discarded.

Conditions

INVREQ
RESP2 values:

33 The terminal is an APPC session or device.
38 The terminal type is neither VTAM nor console.
39 The terminal is local and not out-of-service.
40 The terminal is the system error console.
41 The terminal is an MRO session.
43 Delete processing failed.
44 The terminal is remote and is in use locally.
45 The TERMINAL definition is in use.
46 Discard of this TERMINAL definition is already in progress.
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.

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

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 local CICS system. A transaction class cannot be removed while any TRANSACTION definitions belong to it.

See “Discarding resource definitions” on page 24 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, and class definitions were implicit rather than explicit. For compatibility, CICS supplies definitions for the numbered classes, named ‘DFHTCL*nn*’, where *nn* is the 2-digit class number. You can discard a numbered class by using the associated name for the TRANCLASS value (DFHTCL01 for class 1, for example).

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

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 local 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. Transactions already in flight are not affected; they continue to execute under the definition in force at the time they were attached.

See “Discarding resource definitions” on page 24 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.
13 The transaction is defined in the SIT.
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:

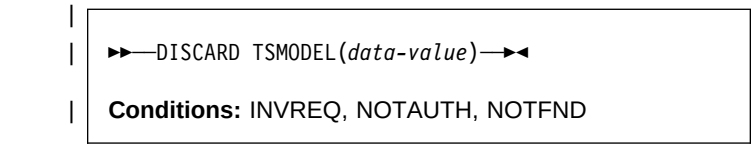
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

TRANSIDERR
RESP2 values:

1 The transaction cannot be found.

DISCARD TSMODEL

Remove a temporary storage model definition.



Description

The DISCARD TSMODEL command removes the definition of a temporary storage model from the local CICS system, so that the system no longer has access to the temporary storage model; that is, it revokes the earlier installation of a TSMODEL resource definition of the same name.

You can discard a TSMODEL, except those begining with DFH, at any time. In-flight UOWs which are using such TSMODELs will complete normally.

See “Discarding resource definitions” on page 24 for general information about discards.

Options

TSMODEL(*data-value***)**
specifies the 8-character name of the temporary storage model that is to be removed.

Temporary storage models whose names begin with the letters DFH are assumed to be CICS-defined temporary storage models and cannot be discarded.

Conditions

INVREQ
RESP2 values:

2 The TSMODEL definition is currently in use.
3 The temporary storage model cannot be discarded because its name begins with DFH.

NOTAUTH
RESP2 values:

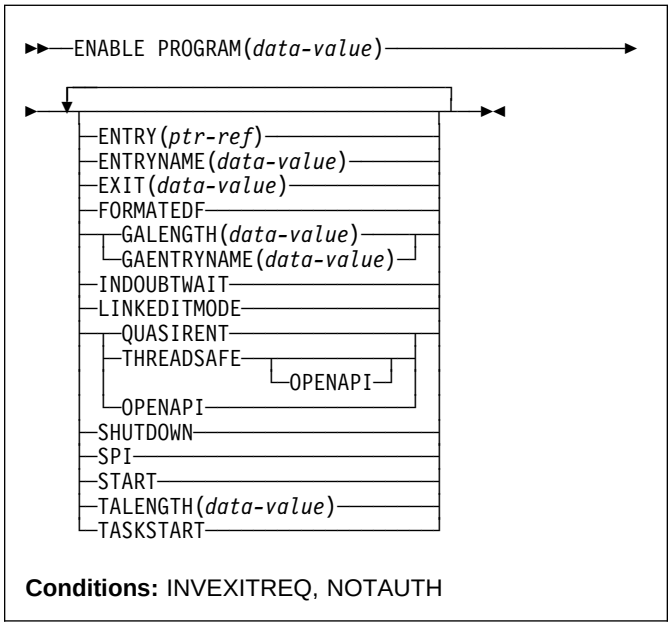
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to discard a TSMODEL definition with this name.

NOTFND
RESP2 values:

1 The TSMODEL does not exist.

ENABLE PROGRAM

Enable a user exit to allow it to be invoked.



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 83 for the correspondence between options on the two commands.

For programming information about exits in CICS, see the *CICS Customization Guide*; you should also see the general discussion of commands that modify exits in “Exit-related commands” on page 25.

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 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 does 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.

INDOUBTWAIT (task-related user exits only)
specifies that the task-related user exit supports the in-doubt protocol. For information about the in-doubt protocol, see the *CICS Customization Guide*.

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.

OPENAPI(task-related user exits only)
This option is accepted and translated by the CICS translator as part of the CICS system programming interface (SPI). However, it is disabled at runtime and has no effect on the task-related user exit program being enabled until the option is enabled in a future release.

When enabled, OPENAPI will specify that the task-related user exit program is using non-CICS APIs. If the user application program that invokes the task-related user exit is defined as quasi-reentrant, CICS will switch the user task to an open TCB before passing control to the task-related user exit program. CICS will assume that a task-related user exit enabled with OPENAPI does not manage its own private pool of TCBs for non-CICS services.

Conversely, if OPENAPI is omitted, CICS will assume that the task-related user exit is either using only the CICS API, or that it performs its own TCB switch to invoke non-CICS services.

Note: If you specify OPENAPI without THREADSAFE, CICS enforces THREADSAFE by default.

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 attempts to build one dynamically if the system is defined to allow autoinstall 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.

QUASIRENT
specifies that the task-related user exit program is quasi-reentrant, and relies on the serialization provided by CICS when accessing shared resources. The task-related user exit program is restricted to the CICS permitted programming interfaces, and must comply with CICS quasi-reentrancy rules. CICS always invokes a quasi-reentrant task-related user exit under the QR TCB. If the task-related user exit program uses MVS services, it must switch to its own private TCB before issuing calls to these services, and switch back again before returning to its caller.

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.

SPI (task-related user exits only)
specifies that the task-related user exit is to be invoked if an INQUIRE EXITPROGRAM command which names it specifies the CONNECTST or QUALIFIER option, or both.

The task-related user exit program is invoked with an SPI call, allowing it to return CONNECTST and QUALIFIER information to the inquiring program. For details of RMI SPI calls, see the *CICS Customization Guide*.

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 is also 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.

THREADSAFE
specifies that the task-related user exit program is written to threadsafe standards, and takes into account the possibility that, when accessing shared resources, other programs may be executing concurrently and attempting to modify the same resources. It uses appropriate serialization techniques when accessing any shared resources.

A threadsafe task-related user exit program must be able to run under whichever TCB CICS invokes it. This could be either the QR TCB or an open TCB. (If OPENAPI is also specified, and when it is enabled, CICS will always invoke the task-related user exit under an L8 open TCB.)

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('EP') 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 loads 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

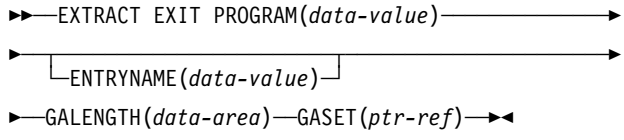
```
EXEC CICS ENABLE PROGRAM('EP9')
     TALENGTH(750) ENTRYNAME('RM1') GALENGTH(200)

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.



Conditions: INVEXITREQ, NOTAUTH

Description

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

Note: To enable the migration of application programs written for earlier releases that specify DSNCEXT1 or DSN2EXT1 on the EXTRACT EXIT command to inquire on the status of the CICS-DB2 interface, CICS automatically substitutes the correct name, DFHD2EX1. CICS does this by setting argument 1 in the parameter list to address the new name, and no application program storage is altered. This allows existing application programs to work unchanged.

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 335.

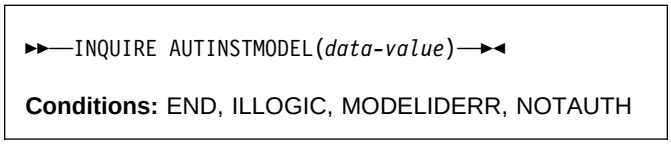
X'800200' The exit is not enabled.
X'800400' The exit has no global work area.
X'808000' The load module named in the PROGRAM option is not the same as the one used when the exit specified in the ENTRYNAME option was enabled.

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.



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 18 for general information about browsing, including 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:

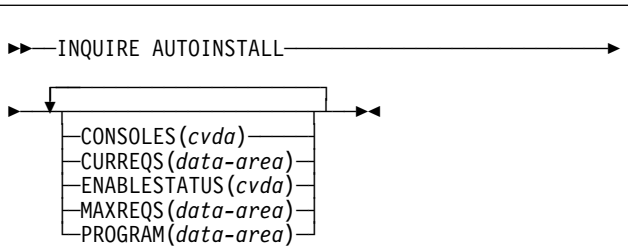
1 The model specified cannot be found.

NOTAUTH
RESP2 values:

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

INQUIRE AUTOINSTALL

Retrieve autoinstall values.



Conditions: NOTAUTH

Description

The INQUIRE AUTOINSTALL returns information relating to the automatic installation (autoinstall) of VTAM terminals, APPC sessions, and MVS consoles in your CICS system.

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

Options

CONSOLES(*cvda*)
returns a CVDA value indicating the status of console autoinstall in CICS. The CVDA values are:

PROGAUTO
Consoles can be autoinstalled if ENABLESTATUS returns a CVDA of ENABLED. The autoinstall control program is called for the install and delete functions.

FULLAUTO
Consoles can be autoinstalled if ENABLESTATUS returns a CVDA of ENABLED. The autoinstall control program is not called for the install and delete functions, and CICS generates the terminal identifier automatically for the consoles it autoinstalls.

NOAUTO
Consoles cannot be autoinstalled.

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 the CICS autoinstall facility. CVDA values are:

DISABLED
Neither consoles nor terminals can be autoinstalled in CICS. DISABLED is returned for the following conditions:

Terminals
MAXREQS equal 0, or the autoinstall control program is disabled.

Consoles
1. CONSOLES CVDA returns NOAUTO.
2. CONSOLES CVDA returns PROGAUTO but autoinstall control program is disabled.

ENABLED
Either consoles or terminals or both can be autoinstalled in CICS. If you want to check whether ENABLED applies to consoles, terminals, or both, check the values returned on other options. ENABLED is returned for the following conditions:

Terminals
MAXREQS not equal 0 and autoinstall control program is enabled.

Consoles
1. CONSOLES CVDA returns FULLAUTO.
2. CONSOLES CVDA returns PROGAUTO and autoinstall control program is enabled.

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. (The MAXREQS option corresponds to the AIQMAX system initialization parameter.)

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.

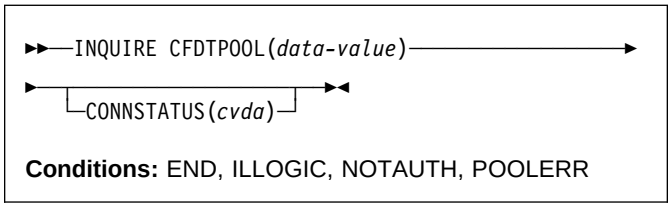
Conditions

NOTAUTH
RESP2 values:

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

INQUIRE CFDTPOOL

Retrieve information about a coupling facility data table pool.



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

Description

The INQUIRE CFDTPOOL command returns the status of the connection of the local CICS region to a coupling facility data table pool.

Browsing

You can also browse through all the coupling facility data table pool names that a CICS region has installed, using the browse options (START, NEXT, and END) on INQUIRE CFDTPOOL commands. CICS implicitly installs coupling facility data table pool names from file definitions that specify a coupling facility data table pool name, even if the definition specifies TABLE(NO). See "Browsing resource definitions" on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

CFDTPOOL(*data-value***)**
specifies the 8-character name of the coupling facility data table pool about which you are inquiring.

CONNSTATUS(*cvda***)**
returns a CVDA value indicating whether CICS is connected to the specified pool.

CVDA values are:

CONNECTED
The server for the coupling facility data table pool is available in this MVS image, and this CICS is currently connected to it.

UNCONNECTED
The server for the coupling facility data table pool is available in this MVS image, but this CICS is not currently connected to it.

UNAVAILABLE
The server for the coupling facility data table pool is currently unavailable in this MVS image.

Conditions

END
RESP2 values:

2 There are no more coupling facility data table pools to browse.

ILLOGIC
RESP2 values:

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

NOTAUTH
RESP2 values:

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

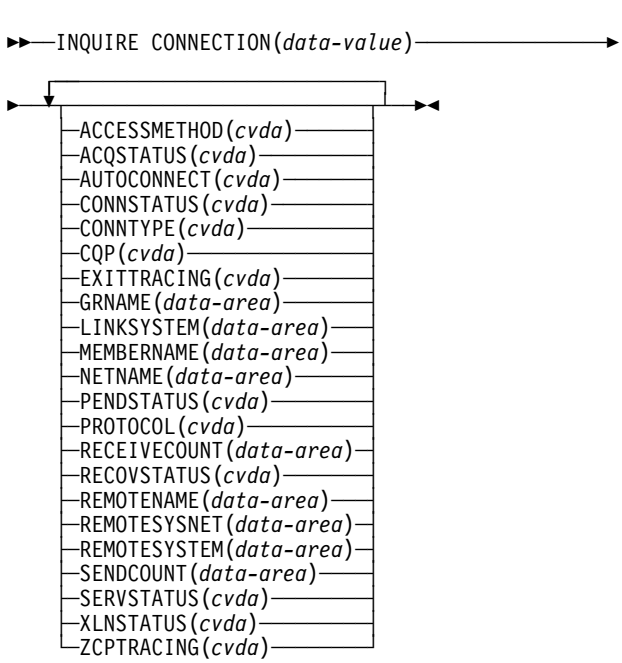
POOLERR
RESP2 values:

1 The named CFDT pool was not found. Either CICS has not installed any file definitions that specify the named coupling facility data table pool, or the name is specified incorrectly on the command.

2 An internal control structure that CICS uses to maintain access to CFDT pools has been altered while the set of pools known to CICS was being browsed.

INQUIRE CONNECTION

Retrieve information about a system connection.



Conditions: END, ILLOGIC, NOTAUTH, SYSIDERR

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.

Remote connections: In addition to links to other systems or devices, some connection definitions refer to remote connections. A remote connection is a link to another system or device that is actually owned by another CICS system. The owning system is called the terminal-owning region (TOR). Note that different connections may have different TORs. TOR here refers to the owning system for a particular connection.

The way that the local system is connected to the TOR makes a difference to the way in which the REMOTESYSTEM and REMOTESYSNET options of the remote CONNECTION definition are specified.

If the TOR is **directly** connected to the local system, the REMOTESYSTEM option usually names the CONNECTION definition for the link. (It can name an indirect connection, but that is an unusual setup). In this case, the netname of the TOR is specified in the link CONNECTION definition. The REMOTESYSNET option of the remote CONNECTION definition may or may not specify the netname of the TOR.

If a remote connection is on a system that is **not directly linked** to the TOR, the REMOTESYSTEM option can name one of two types of connection, as follows:

- A “real” connection that is the next link in the chain towards the TOR. In this case, the REMOTESYSNET option must specify the netname of the TOR. For example, in Figure 2 on page 108, connection CON2 points to connection INTS, which is the first link in the chain to the TOR.
- An indirect connection. In this case, the indirect connection NETNAME contains the netname of the TOR, and its INDSYS option names another connection, which can also be indirect or “real.” For example, in Figure 2 on page 108 , connection CON1 points to connection INDC, which in turn points to INTS. The REMOTESYSNET option of the remote CONNECTION definition may or may not specify the netname of the TOR.

Note, however, that for remote connection definitions:

- The LINKSYSTEM option of INQUIRE CONNECTION returns the “real” connection that is the next link towards the TOR. It is determined by looking at the logical chain of entries from the connection in question to the “real” connection entry. If the chain is broken (because an entry has not been installed yet, or has been discarded) LINKSYSTEM is not set.
- The REMOTESYSNET option of INQUIRE CONNECTION always returns the netname of the TOR (even if the REMOTESYSNET option was not specified on the original CONNECTION definition), unless both the following are true:
  - The system returned in the REMOTESYSTEM field has not been installed.
  - REMOTESYSNET was not specified on the CONNECTION definition.

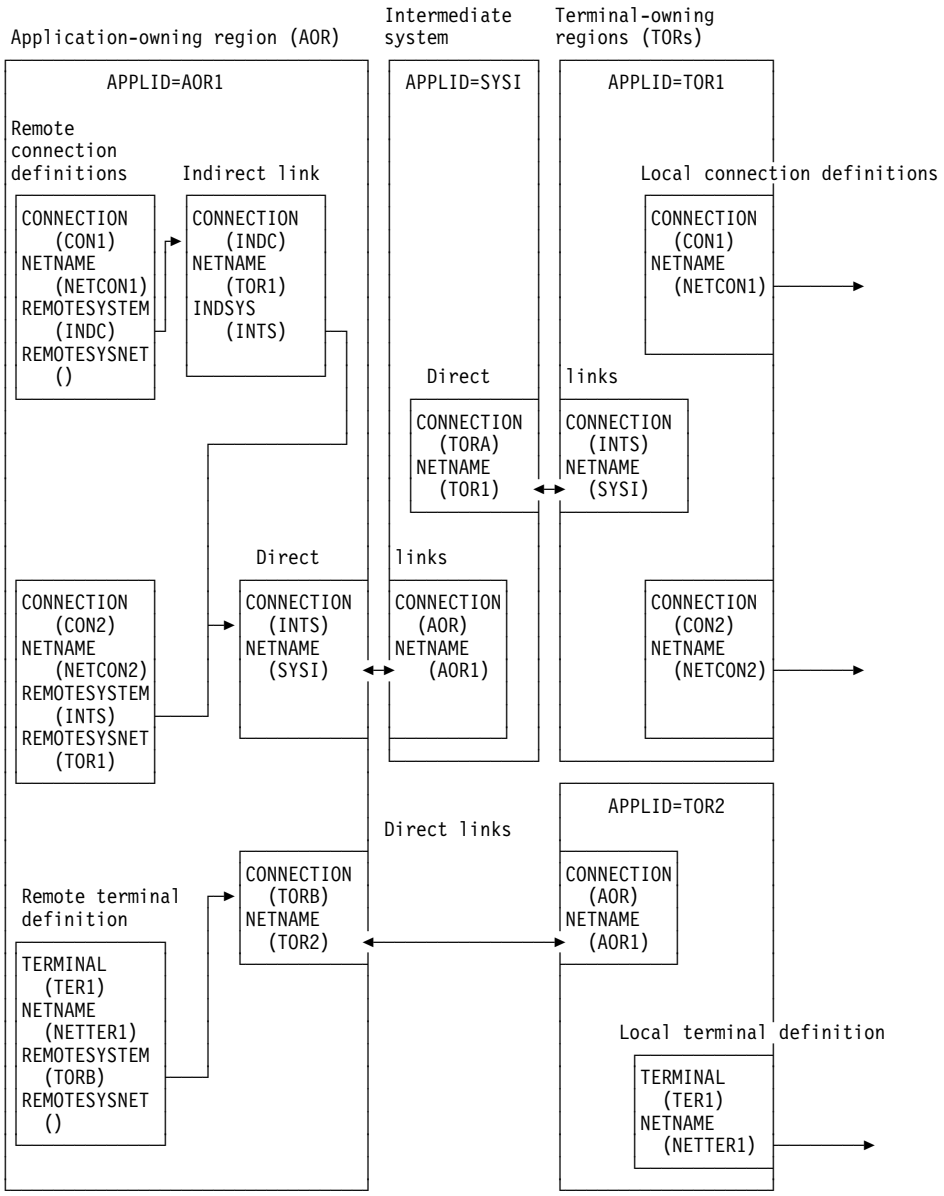


Figure 2. Remote definitions. How the REMOTESYSTEM and REMOTESYSNET options of the CONNECTION definition are specified depends on the path to the TOR. The LINKSYSTEM field of INQUIRE CONNECTION always returns the real link sysid. The REMOTESYSNET field of INQUIRE CONNECTION always returns the netname of the TOR, if it exists.

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 18 for general information about browsing, including 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 it is not, the XCF access method is 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 it is not, the XCF access method is 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 (APPC only)
: The connection is acquired but there are currently no bound sessions because they were unbound for limited resource reasons.

FREEING (APPC only)
: The connection is being released.

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

OBTAINING (APPC only)
: 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.

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.

NOTAPPLIC
The connection is not an EXCI connection.

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

See the *Internet and External Interfaces Guide* for more information about EXCI connections.

CQP(*cvda***)**
returns a CVDA indicating the status of the connection quiesce protocal for the connection. The CVDA values are:

COMPLETE The quiesce protocol completed successfully when the connection was released. This reverts to NOTATTEMPTED if the connection is reacquired.

FAILED The protocol failed. This can occur for one of several reasons, such as a session failure during execution of the protocol, or because the partner receiving the CQP flow has outstanding work.

NOTATTEMPTED The connection supports the protocol, but it has not yet been invoked because the connection status is ACQUIRED.

NOTSUPPORTED The connection does not support the quiesce protocol. This could be, for example, because the partner is a back-level CICS region that does not support the connection quiesce protocol.

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.

GRNAME(*data-area***)**
returns (for an APPC connection to a generic resource when this system is also a generic resource) the 8-character generic resource name of the connected LU. Otherwise it returns blanks. CICS assumes that the partner is a generic resource if the two NETNAMEs sent with a BIND are different. This information may also be returned for a partner which is not a generic resource but which uses XRF.

LINKSYSTEM(*data-area***)**
returns the 4-character name of the connection that is the real link towards the TOR for a remote or indirect system entry, if it is available. It is not set if some connection definitions in the chain from the remote or indirect entry to the link system are missing.

MEMBERNAME(*data-area***)**
returns (for an APPC connection to a generic resource when this system is also a generic resource) the 8-character member name (applid) of the connected LU. Otherwise it returns blanks. CICS assumes that the partner is a generic resource if the two NETNAMEs sent with a BIND are different. This information may also be returned for a partner that is not a generic resource but which uses XRF.

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 and MRO only)**
returns a CVDA value identifying whether there are any pending units of work for this connection. CVDA values are:

NOTAPPLIC
This is not an APPC parallel-session nor a CICS-to-CICS MRO connection.

NOTPENDING
There has been no mismatch of lognames with the partner.

Note: MRO connections to pre-CICS Transaction Server for OS/390 systems do not use lognames. Therefore, for these connections, PENDSTATUS always returns NOTPENDING.

PENDING
There is resynchronization work outstanding for the connection but the partner system has performed an initial start, preventing completion of the resynchronization process. (If the partner is a pre-CICS Transaction Server for OS/390 system, its performing a cold start has the same effect.) The units of work associated with the connection may need to be individually investigated and forced to commit or backout. Alternatively, the units of work can be unilaterally committed or backed out, according to their associated transaction definitions, by using the SET CONNECTION NOTPENDING command.

If this is an APPC connection, no new syncpoint work (that is, work involving synclevel 2 protocols) can be transmitted across it until a SET CONNECTION NOTPENDING command has been issued. This restriction does not apply to MRO connections.

If you are not concerned by the loss of synchronization caused by the initial (or cold) start of the partner, you can cause the SET CONNECTION NOTPENDING command to be issued automatically by specifying XLNACTION(FORCE) on the CONNECTION definition.

For further information about pending units of work, see the *CICS 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.

RECOVSTATUS(cvda) (APPC and MRO only)
returns a CVDA value indicating whether there is resynchronization work outstanding for the connection. The connection may never have been connected, have been quiesced and all resynchronization work completed, or disrupted without quiesce—in which case resynchronization may be necessary. CVDA values are:

NORECOVDATA
Neither side has recovery information outstanding.

NOTAPPLIC
This is not an APPC parallel-session nor a CICS-to-CICS MRO connection, and does not support two-phase commit protocols.

NRS CICS does not have recovery outstanding for the connection, but the partner may have.

RECOVDATA
There are in-doubt units of work associated with the connection, or there are outstanding resyncs awaiting FORGET on the connection. Resynchronization takes place when the connection next becomes active, or when the UOW is unshunted.

If there is recovery outstanding, then on completion of exchange lognames, either resynchronization takes place or, in the case of a cold exchange, the PENDING condition is created.

REMOTENAME(*data-area*)
returns the 4-character name by which this connection is known in a remote system, if the subject of the inquiry is a remote connection.

REMOTESYSNET(*data-area*)
returns the 8-character netname of the owning TOR, if the subject of this inquiry is a remote connection. If it is blank, but the connection is remote, the system named in the REMOTESYSTEM field has not been installed, and no value was specified for the REMOTESYSNET option when the connection was defined.

REMOTESYSTEM(*data-area*)
returns the 4-character name of a connection, if the subject of the inquiry is a remote connection. The named connection can be either a connection entry that links towards the TOR, or an indirect connection which provides the netname of the TOR, and itself points to another connection.

Otherwise this field is blank.

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 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:

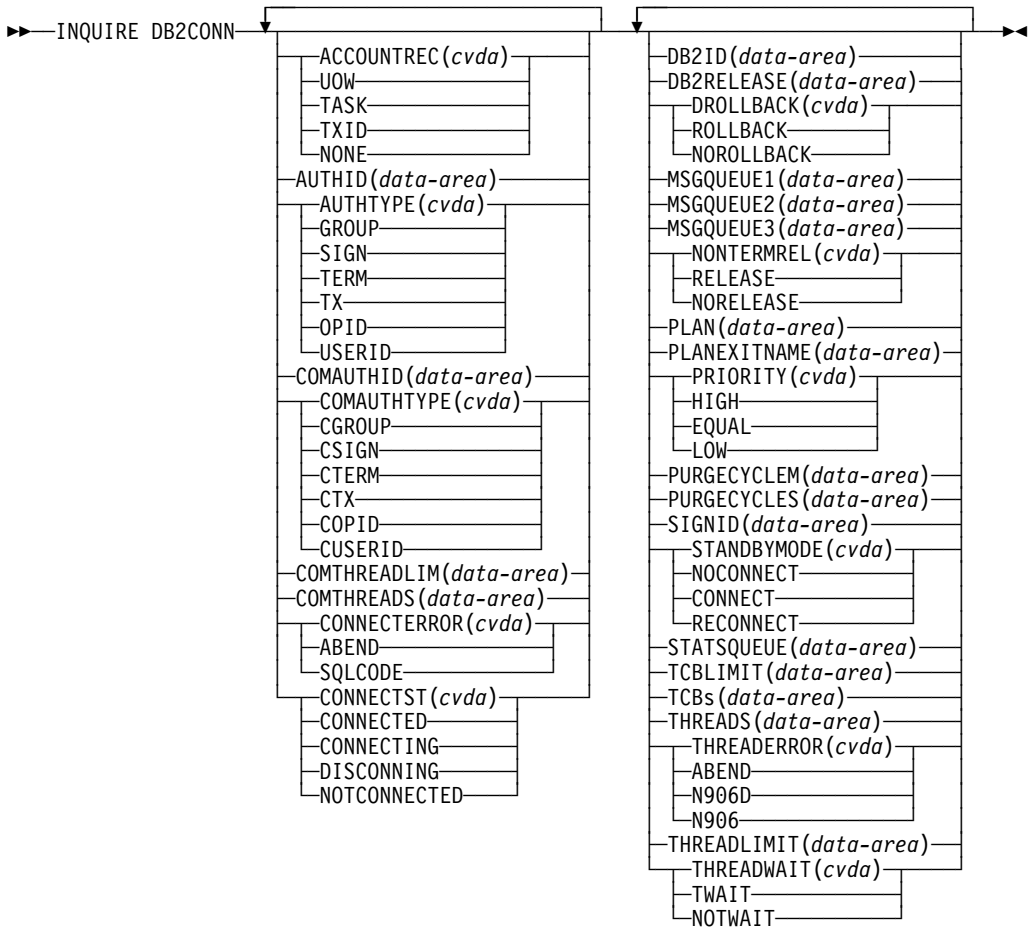
100 The user associated with the issuing task is not authorized to use this command.

SYSIDERR
RESP2 values:

1 The connection cannot be found.

INQUIRE DB2CONN

Defines the attributes of the connection made between CICS and DB2.



Conditions: NOTAUTH, NOTFND

Description

The INQUIRE DB2CONN command allows you to inquire about attributes of the currently installed DB2CONN which defines the connection to DB2.

Note that because there can be only one DB2CONN installed at a time, the name of the DB2CONN is not required on input.

Options

ACCOUNTREC
returns the minimum amount of DB2 accounting required for transactions using pool threads. The specified minimum may be exceeded as described in the following options. CVDA values are:

UOW
The CICS DB2 attachment facility causes an accounting record to be produced by DB2 for each UOW, assuming that the thread is released at the end of the UOW.

TASK
The CICS DB2 attachment facility causes a minimum of one accounting record to be produced by DB2 for each CICS task.

A transaction containing multiple UOWs (assuming the thread is released at syncpoint) may use a different thread for each of its UOWs. The result may be the production of an accounting record for each UOW. For example, an accounting record is produced if a thread terminates after being released, or if a thread is reused but the primary AUTHID is changed.

TXID
: The CICS DB2 attachment facility causes an accounting record to be produced by DB2 when the transaction ID using the thread changes.

  Because pool threads are typically used by a number of different transaction IDs, there is an increased chance that a transaction containing multiple UOWs will use a different thread for each UOW (assuming the thread is released at syncpoint). In this case an accounting record may be produced for each UOW.For example, an accounting record is produced if a thread terminates after being released, or if a thread is reused but the primary AUTHID is changed.

NONE
: No accounting records are required for transactions using pool threads.

  DB2 nevertheless produces at least one accounting record for each thread when the thread is terminated. Additionally, authorization changes cause accounting records to be produced.

AUTHID
returns an id to be used for security checking when using pool threads. If an AUTHID is returned, AUTHTYPE is not applicable.

AUTHTYPE
returns the type of id to be used for security checking when using pool threads. If an AUTHType is returned, AUTHid is blank. CVDA values are:

GROUP
: The 8-character USERID and the connected group name are used as the authorization ID. The following table shows how these two values are interpreted by DB2.

	IDs passed to DB2	How DB2 interprets values
	CICS sign-on user ID (USERID)	Represents the primary DB2 authorization ID.
	RACF-connected group name	If the RACF list of group options is not active, DB2 uses the connected group name supplied by the CICS attachment facility as the secondary DB2 authorization ID. If the RACF list of group options is active, DB2 ignores the connected group name supplied by the CICS attachment facility, but the value appears in the DB2 list of secondary DB2 authorization IDs.

  If no RACF group ID is available for this USERID, an 8-character field of blanks is passed to DB2 as the group ID.

SIGN
: The SIGNID parameter of the DB2CONN is used as the resource authorization ID.

TERM
: The terminal identification (four characters padded to eight) is used as an authorization ID. An authorization ID cannot be obtained in this manner if a terminal is not connected with the transaction.

  If a transaction is started (using a CICS command) and has no terminal associated with it, AUTHTYPE(TERM) should not be used.

TX
: The transaction identification (four characters padded to eight) is used as the authorization ID.

OPID
: The user operator identification associated with the userid, associated with the CICS transaction, is used as the authorization ID (three characters padded to eight).

USERID
: The 8-character USERID associated with the CICS transaction is used as the authorization ID.

  When the DB2 sample sign-on exit DSN3@SGN is used with AUTHTYPE(USERID), the exit sends the USERID to DB2 as the primary authorization ID and the RACF group ID to DB2 as the secondary ID. When the sample sign-on exit is used, there is no difference between AUTHTYPE(USERID) and AUTHTYPE(GROUP).

COMAUTHID
returns an ID to be used for security checking when using command threads. If COMAUTHType is returned, COMAUTHid is not applicable.

COMAUTHTYPE
returns the type of ID to be used for security checking when using command threads. If COMAUTHType is returned, COMAUTHid is blank. CVDA values are:

CGROUP
: The 8-character USERID and the connected group name are used as the authorization ID. The following table shows how these two values are interpreted by DB2.

	IDs passed to DB2	How DB2 interprets values
	CICS sign-on user ID (USERID)	Represents the primary DB2 authorization ID.
	RACF-connected group name	If the RACF list of group options is not active, DB2 uses the connected group name supplied by the CICS attachment facility as the secondary DB2 authorization ID. If the RACF list of group options is active, DB2 ignores the connected group name supplied by the CICS attachment facility, but the value appears in the DB2 list of secondary DB2 authorization IDs.

If no RACF group ID is available for this USERID, an 8-character field of blanks is passed to DB2 as the group ID.

CSIGN
: The SIGNID parameter of the DB2CONN is used as the resource authorization ID.

CTERM
: The terminal identification (four characters padded to eight) is used as an authorization ID. An authorization ID cannot be obtained in this manner if a terminal is not connected with the transaction.

  If a transaction is started (using a CICS command) and has no terminal associated with it, COMAUTHTYPE(CTERM) should not be used.

CTX
: The transaction identification (four characters padded to eight) is used as the authorization ID.

COPID
: The operator identification associated with the userid that is associated with the CICS transaction is used as the authorization ID (three characters padded to eight).

CUSERID
: The 8-character USERID associated with the CICS transaction is used as the authorization ID.

  When the DB2 sample sign-on exit DSN3@SGN is used with COMAUTHTYPE(CUSERID), the exit sends the USERID to DB2 as the primary authorization ID and the RACF group ID to DB2 as the secondary ID. When the sample sign-on exit is used, there is no difference between COMAUTHTYPE(CUSERID) and COMAUTHTYPE(CGROUP).

COMTHREADS
: returns the current number of active command threads.

COMTHREADLIM
: returns the current maximum number of command threads the CICS DB2 attachment allows active before requests overflow to the pool.

CONNECTERROR
: returns how the fact that CICS is not connected to DB2 because the adapter is in 'standby mode' is reported back to an application that has issued a SQL request. CVDA values are:

  ABEND
  : The application is abended with abend AEY9.

  SQLCODE
  : The application receives a -923 SQLCODE.

CONNECTST
: returns the status of the CICS DB2 connection. CVDA values are:

  CONNECTED
  : CICS is connected to DB2.

  NOTCONNECTED
  : CICS is not connected to DB2.

  CONNECTING
  : CICS is currently attempting to connect to DB2.

  DISCONNING
  : CICS is currently disconnecting from DB2.

DB2ID
: returns the name of the DB2 subsystem that the CICS DB2 attachment is connected to or is to connect to.

DB2RELEASE
: returns a four-character value indicating the version and release level of the DB2 subsystem CICS is connected to. When CICS is not connected to DB2, blanks are returned.

DROLLBACK
: returns a value showing whether the CICS DB2 attachment is to initiate a SYNCPOINT ROLLBACK in the event of a transaction being selected as victim of a deadlock resolution. CVDA values are:

  ROLLBACK
  : the attachment facility issues a sync point rollback before returning control to the application. An SQL return code of -911 is returned to the program.

  NOROLLBACK
  : the attachment facility is not to initiate a rollback for a transaction. An SQL return code of -913 is returned to the application.

MSGQUEUE1
: returns the name of the first transient data destination to which unsolicited messages from the CICS DB2 attachment are sent.

MSGQUEUE2
: returns the name of the second transient data destination to which unsolicited messages from the CICS DB2 attachment are sent.

MSGQUEUE3
returns the name of the third transient data destination to which unsolicited messages from the CICS DB2 attachment are sent.

NONTERMREL
retuns a value showing whether non-terminal transactions are to release threads for reuse at intermediate syncpoints. CVDA values are:

RELEASE
non-terminal transactions release threads for reuse at intermediate syncpoints.

NORELEASE
non-terminal transactions do not release threads for reuse at intermediate syncpoints.

PLAN
returns the name of the plan used for the pool. If a plan name is returned, PLANEXITNAME is blank,

PLANEXITNAME
returns the name of the dynamic plan exit used for pool threads. If a PLANEXITNAME is returned, PLAN is blank,

PRIORITY
returns the priority of the pool thread subtasks relative to the CICS main task (QR TCB). CVDA values are:

HIGH
subtasks attain a higher priority than the CICS main task from which the subtask was generated.

EQUAL
subtasks have equal priority with the CICS main task.

LOW
subtasks have a lower priority than the CICS main task.

PURGECYCLEM
returns in minutes the length of the protected thread purge cycle. The range for PURGECYCLEM is 0-59.

A protected thread is not terminated immediately when it is released. It is terminated only after two completed purge cycles, if it has not been reused in the meantime. Hence if the purge cycle is set to 30 seconds after it is released, a protected thread is purged 30 - 60 seconds after it is released. An unprotected thread is terminated when it is released (at syncpoint or end of task) if there are no other transactions waiting for a thread on that DB2ENTRY.

PURGECYCLES
returns in seconds the length of the protected thread purge cycle. The range for PUrgecycles is 30-59.

A protected thread is not terminated immediately when it is released. It is terminated only after two completed purge cycles, if it has not been reused in the meantime. Hence if the purge cycle is set to 30 seconds after it is released, a protected thread is purged 30 - 60 seconds after it is released. An unprotected thread is terminated when it is released (at syncpoint or end of task) if there are no other transactions waiting for a thread on that DB2ENTRY.

SIGNID
returns the authorization ID to be used by the CICS DB2 attachment when signing on to DB2 for pool and DB2 entry threads specifying AUTHTYPE(SIGN) and command threads specifying COMAUTHTYPE(CSIGN).

STANDBYMODE
returns the action to be taken by the CICS DB2 attachment if DB2 is not active when an attempt is made to start the connection from CICS to DB2. CVDA values are:

NOCONNECT
The CICS DB2 attachment terminates.

CONNECT
The CICS DB2 attachment goes into 'standby mode' to wait for DB2.

RECONNECT
The CICS DB2 attachment goes into 'standby mode' and waits for DB2. Having connected to DB2, if DB2 subsequently fails the CICS DB2 attachment reverts to standby mode again and subsequently reconnects to DB2 when it comes up again.

STATSQUEUE
returns the transient data destination for CICS DB2 attachment statistics produced when the CICS DB2 attachment is shutdown.

TCBs
returns the current number of subtask TCBs attached to service command, pool, and DB2ENTRY threads.

TCBLIMIT
returns the current maximum number of subtasks that can be identified to DB2.

THREADERROR
returns the processing that is to occur following a create thread error. CVDA values are:

ABEND
For a second or subsequent SQL error the transaction is abended with abend code AD2S, AD2T, or AD2U, dependent on the type of error that occurred. The transaction must be terminated and reinitialized before it is allowed to issue another SQL request.

N906D
A transaction dump is to be taken and the DSNCSQL RMI associated with the transaction is *not* to be disabled. The transaction receives a -906 SQLCODE if another SQL is issued, unless the transaction issues SYNCPOINT ROLLBACK. SYNCPOINT without the ROLLBACK option results in an ASP3 or ASP7 abend. The

transaction dump records an abend of AD2S, AD2T, or AD2U.

N906
: The DSNCSQL RMI associated with the transaction is *not* to be disabled. The transaction receives a -906 SQLCODE if another SQL request is issued, unless the transaction issues a SYNCPOINT ROLLBACK. SYNCPOINT without the ROLLBACK option results in an ASP3 or ASP7 abend.

THREADLIMIT
: returns the current maximum number of pool threads the CICS DB2 attachment allows active before requests are made to wait or are rejected (see THREADWait).

THREADS
: returns the current number of active pool threads.

THREADWAIT
: returns whether or not transactions should wait for a pool thread or be abended should the number of active pool threads reach the threadlimit number. CVDA values are:

  TWAIT
  : If all threads are busy, a transaction waits until one becomes available.

  NOTWAIT
  : If all threads are busy, a transaction is terminated with an abend code AD3T.

Conditions

NOTFND
: RESP2 values:

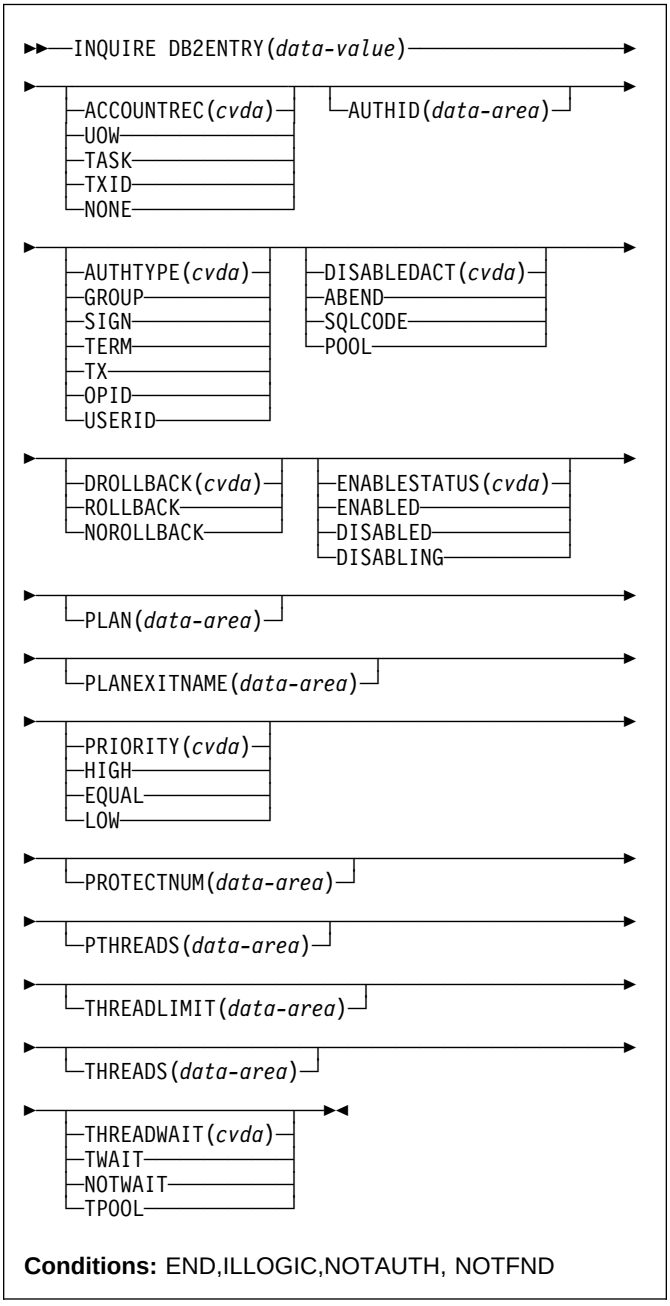
  1 The DB2CONN cannot be found.

NOTAUTH
: RESP2 values:

  100 Command authorization failure

INQUIRE DB2ENTRY

Returns the attributes of the DB2ENTRY that defines resources to be used by a specific transaction or by a group of transactions when accessing DB2.



Conditions: END,ILLOGIC,NOTAUTH, NOTFND

Description

| The entry is identified by the name it was defined with in the
| CSD by the DEFINE DB2ENTRY command. For RCTs
| migrated to the CSD, it is identified by the name of the first transaction on the DSNCRCT TYPE=ENTRY statement unless the RDONAME parameter has been specified.

Browsing

You can also browse through all of the DB2ENTRYs installed in a CICS region by using the browse options (START, NEXT, and END) on INQUIRE DB2ENTRY commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

ACCOUNTREC
returns the minimum amount of DB2 accounting required for transactions using this DB2ENTRY. The specified minimum may be exceeded, as described in the following options. CVDA values are:

UOW
The CICS DB2 attachment facility causes an accounting to be produced by DB2 for each UOW, assuming that the thread is released at the end of the UOW.

TASK
The CICS DB2 attachment facility causes a minimum of one accounting record to be produced by DB2 for each CICS task.

A transaction containing multiple UOWs may use a different thread for each UOW (assuming the thread is released at syncpoint). The result may be the production of an accounting record for each UOW. For example, an accounting record is produced if a thread terminates after being released, or if a thread is reused but the primary AUTHID is changed.

TXID
The CICS DB2 attachment facility causes an accounting record to be produced by DB2 when the transid using the thread changes.

This option applies to DB2ENTRYs that are used by more than one transaction ID. As threads are typically released at syncpoint, a transaction containing multiple UOWs may use a different thread for each UOW. The result may be that an accounting record is produced for each UOW. For example, an accounting record is produced if a thread terminates after being released, or if a thread is reused but the primary AUTHID is changed.

NONE
No accounting records are required for transactions using threads from this DB2ENTRY.

DB2 produces. however, at least one accounting record per thread when the thread is terminated. Additionally, authorization changes cause accounting records to be produced.

AUTHID
returns an id to be used for security checking for threads on this DB2ENTRY. If an AUTHId is returned, AUTHType is not applicable.

AUTHTYPE
returns the type of id to be used for security checking for threads on this DB2ENTRY. If an AUTHType is returned, AUTHid is blank. CVDA values are:

GROUP
The 8-character userid and the connected group name are used as the authorization ID. The following table shows how these two values are interpreted by DB2.

	IDs passed to DB2	How DB2 interprets values
	CICS sign-on user ID (USERID)	Represents the primary DB2 authorization ID.
	RACF-connected group name	If the RACF list of group options is not active, then DB2 uses the connected group name supplied by the CICS attachment facility as the secondary DB2 authorization ID. If the RACF list of group options is active, DB2 ignores the connected group name supplied by the CICS attachment facility, but the value appears in the DB2 list of secondary DB2 authorization IDs.

If no RACF group ID is available for this USERID, then an 8-character field of blanks is passed to DB2 as the group ID.

SIGN
The SIGNID parameter of the DB2CONN is used as the resource authorization ID.

TERM
The terminal identification (four characters padded to eight) is used as an authorization ID. An authorization ID cannot be obtained in this manner if a terminal is not connected with the transaction.

If a transaction is started (using a CICS command) and has no terminal associated with it, AUTHTYPE(TERM) should not be used.

TX The transaction identification (four characters padded to eight) is used as the authorization ID.

OPID
The operator identification associated with the userid that is associated with the CICS transaction is used as the authorization ID (three characters padded to eight).

USERID
The 8-character USERID associated with the CICS transaction is used as the authorization ID.

When the DB2 sample sign-on exit DSN3@SGN is used with AUTHTYPE(USERID), the exit sends

the USERID to DB2 as the primary authorization ID and the RACF group ID to DB2 as the secondary ID. When the sample sign-on exit is used, there is no difference between AUTHTYPE(USERID) and AUTHTYPE(GROUP).

DISABLEDACT
returns what CICS is to do with new transactions accessing the DB2ENTRY when it has been disabled or disabling. If DISABLEDACT is not specified, and DB2ENTRY is disabled, new requests are routed to the pool by default. CVDA values are:

POOL
The CICS DB2 attachment facility routes the request to the pool. Message DFHDB2072 is sent to the transient data destination specified by MSGQUEUEn on the DB2CONN for each transaction routed to the pool.

ABEND
The CICS DB2 attachment facility abends the transaction.The abend code is AD26.

SQLCODE
An SQLCODE is returned to the application indicating that the DB2ENTRY is disabled.

DROLLBACK
returns whether or not the CICS DB2 attachment should initiate a SYNCPOINT rollback in the event of a transaction being selected as victim of a deadlock resolution. CVDA values are:

ROLLBACK
The attachment facility issues a sync point rollback before returning control to the application. An SQL return code of -911 is returned to the program.

NOROLLBACK
The attachment facility is not to initiate a rollback for this transaction. An SQL return code of -913 is returned to the application.

ENABLESTATUS
returns a cvda indicating whether the DB2ENTRY can be accessed by applications. CVDA values are:

ENABLED
The DB2ENTRY can be accessed by applications. DB2ENTRY is installed in an ENABLED state.

DISABLED
The DB2ENTRY cannot be accessed by applications.

DISABLING
The DB2ENTRY is in the process of being disabled. New transactions cannot access the DB2ENTRY. Existing transactions using the DB2ENTRY are allowed to complete unless the DB2ENTRY is being disabled with the FORCE option.

PLAN
returns the name of the plan to be used for this DB2ENTRY. If PLAN is returned, PLANEXITNAME is blank.

PLANEXITNAME
returns the name of the dynamic plan exit (if any) to be used for this DB2ENTRY. If PLANEXITname is returned, PLAN is blank.

PRIORITY
returns the priority of the thread subtasks for this DB2ENTRY relative to the CICS main task (QR TCB). CVDA values are:

HIGH
subtasks attain a higher priority than the CICS main task from which the subtask was generated.

EQUAL
subtasks have equal priority with the CICS main task.

LOW
subtasks have a lower priority than the CICS main task.

PROTECTNUM
returns the maximum number of protected threads allowed for this DB2ENTRY.

PTHREADS
returns the current number of protected threads for this DB2ENTRY. A protected thread is an inactive thread available for reuse by a new transaction. If no transaction has reused the thread by the time it has been processed by 2 purge cycles, the thread is terminated.

THREADS
returns the current number of threads active for this DB2ENTRY.

THREADLIMIT
returns the current maximum number of threads for this DB2ENTRY that the CICS DB2 attachment allows active before requests are made to wait, overflow to the pool, or are rejected (see THREADWait).

THREADWAIT
returns whether or not transactions should wait for a DB2ENTRY thread be abended, or overflow to the pool if the number of active DB2ENTRY threads reach the Threadlimit number. CVDA values are:

TWAIT
If all threads are busy, a transaction waits until one becomes available.

NOTWAIT
If any threads are busy, a transaction is terminated with an abend code AD2P.

TPOOL
If all threads are busy, a transaction is diverted to use a pool thread. If the pool is also busy, and

NOTWAIT has been specified for the THREADWAIT parameter on the DB2CONN, the transaction is terminated with an abend code AD3T.

Conditions

NOTFND
RESP2 values:

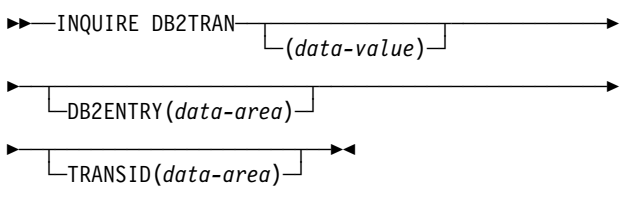
1 The DB2ENTRY cannot be found.

NOTAUTH
RESP2 values:

100 Command authorization failure
101 Resource authorization failure

INQUIRE DB2TRAN

Returns attributes of a particular DB2TRAN which associates a transaction or group of transactions with a DB2ENTRY.



Conditions: END,ILLOGIC,NOTAUTH, NOTFND

Description

The DB2TRAN is identified by the name it was defined with in CEDA. For RCTs migrated to the CSD, the name is the same as the transaction for which the DB2TRAN is being created.

If a TRANSID is specified on a DB2ENTRY when the DB2ENTRY is installed, CICS installs a DB2TRAN named DFHtttt, where tttt is the TRANSID.

Browsing

You can also browse through all of the DB2TRANs installed in your system by using the browse options (START, NEXT, and END) on INQUIRE DB2TRAN commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

DB2ENTRY
returns the name of the DB2Entry to which this DB2Tran refers; that is, the DB2ENTRY with which this additional transaction is associated.

TRANSID
specifies the transaction id to be associated with the entry. The transaction id can include wildcard characters (see the *CICS Resource Definition Guide* for information about use of wildcard characters).

Conditions

NOTFND
RESP2 values:

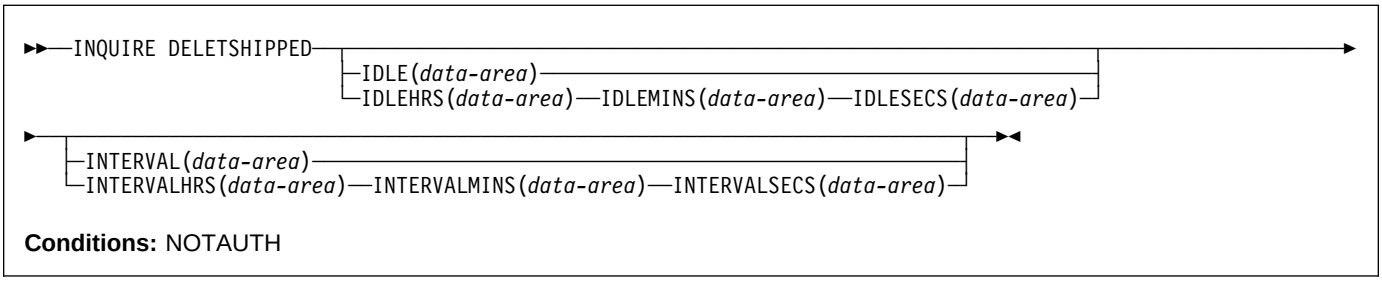
1 The DB2TRAN cannot be found.

NOTAUTH
RESP2 values:

100 Command authorization failure
101 Resource authorization failure

INQUIRE DELETSHIPPED

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



Conditions: NOTAUTH

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 DELETSHIPPED 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.

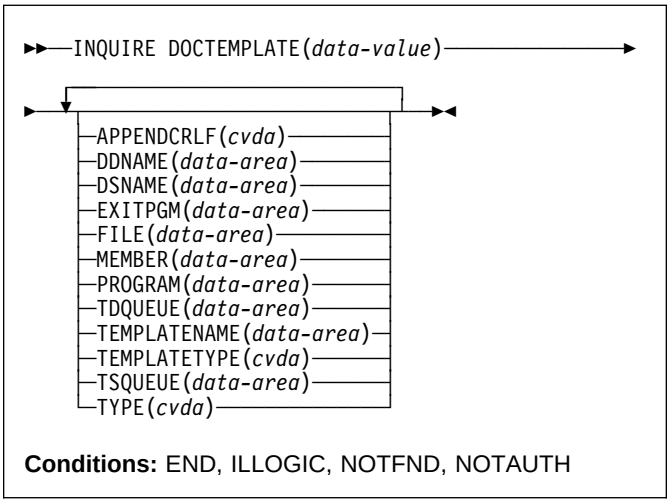
Conditions

NOTAUTH
RESP2 values:

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

INQUIRE DOCTEMPLATE

Find out whether a DOCTEMPLATE is installed.



Conditions: END, ILLOGIC, NOTFND, NOTAUTH

Description

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

Browsing

You can also browse through all of the DOCTEMPLATE installed in your system by using the browse options (START, NEXT, and END) on INQUIRE DOCTEMPLATE commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

APPENDCRLF(*cvda***)**
returns a CVDA value identifying whether CICS is to delete trailing blanks from and append carriage-return line-feed to each logical record of the template. CVDA values are:

APPEND
delete trailing blanks from and append carriage-return line-feed to each logical record of the template.

NOAPPEND
do not delete trailing blanks from or append carriage-return line-feed to each logical record of the template.

DDNAME(*data-value***)**
returns the DD name of the PDS containing the template. The DDName applies only to a template of type PDS.

DOCTEMPLATE(*data-value***)**
specifies the 8-character identifier of the DOCTEMPLATE about which you are inquiring.

DSNAME(*data-value***)**
returns the data set name of the PDS containing the template. It applies only to a template of type PDS.

EXITPGM(*data-value***)**
returns the exit program to be invoked when a request is made for this template. The exit program is passed an architected commarea containing the address and length of a buffer into which the exit program returns the template.

FILE(*data-value***)**
returns the 8-character name of the CICS file definition for the data set containing the template.

MEMBER(*data-value***)**
returns the name of the member in the PDS containing the template. MEMBER applies only to a template of type PDS.

PROGRAM(*data-value***)**
returns the program in which the template data is stored. CICS loads the program and takes all data after the entrypoint to be the template.

TDQUEUE(*data-value***)**
returns the name of the TD queue on which the template is stored.

TEMPLATENAME(*data-value***)**
returns the extended template-name by which the doctemplate is to be known outside the resource definition function.

TEMPLATETYPE(*cvda***)**
returns a CVDA value identifying the type of the source of this template. CVDA values are:

DSNAME

EXITPGM

FILE

PDS

PROGRAM

TDQUEUE

TSQUEUE

TSQUEUE(*data-value***)**
returns the name of the TS queue on which the template is stored.

TYPE(*data-value***)**
returns a CVDA value identifyign the format of the template contents. CVDA values are:

BINARY

EBCDIC

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:

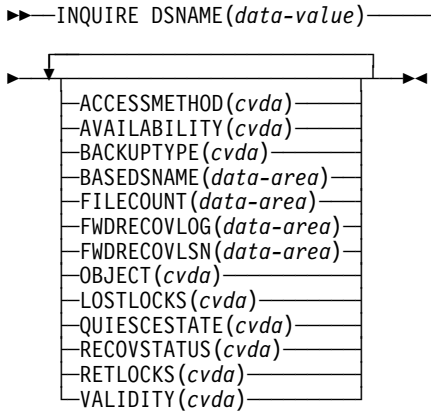
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The DOCTEMPLATE specified cannot be found.

INQUIRE DSNAME

Retrieve information about an external data set.



Conditions: DSNNOTFOUND, END, ILLOGIC, IOERR, NOTAUTH

For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” 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. Many of the attributes of a data set cannot be determined until the first file that references the data set has been opened by the CICS region in which the command is issued. Where an attribute is not valid until a file has been opened, the NOTAPPLIC state is returned.

Note: Using options that require a read from the ICF catalog can slow down the processing of this command.

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 18 for general information about browsing, including 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.

NOTAPPLIC
The data set has not been opened by the CICS region in which the command is issued.

VSAM
The access method is VSAM.

AVAILABILITY(*cvda*) (VSAM only)
returns a CVDA value indicating whether the data set is currently flagged, in this CICS region, as available or unavailable for use. The availability indicator is a local flag that a CICS region maintains in a data set name block (DSNB) for each data set. CVDA values are:

AVAILABLE
The data set is available for use according to the CICS data set name block. CICS can issue both RLS and non-RLS open requests for this data set.

Note: Although a data set is available according to information held by CICS, an open request could still fail if the ICF catalog indicates otherwise. This can occur, for example, if data set recovery is pending or actually in progress.

NOTAPPLIC
The data set is not a VSAM data set, or the data set has not been opened by the CICS region in which this command is issued.

UNAVAILABLE
Returned for a data set that CICS has marked as not available for use. The CICS region is unable to open the data set in either RLS or non-RLS mode.

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. The data set is eligible for BWO and it is accessed in non-RLS mode.

If the data set is opened in RLS mode, you need to look in the VSAM catalog to find out whether the data set is eligible for BWO. NOTAPPLIC is returned as the BACKUPTYPE for data sets opened RLS mode.

NOTAPPLIC
The data set has not been opened by the CICS region in which the command is issued, or the data set is BDAM or a VSAM PATH. Also, if the data set has been opened in RLS mode, NOTAPPLIC is returned. The VSAM catalog should be referred to get the BWO status.

STATIC
The data set is accessed in non-RLS mode, and is not eligible for BWO processing. All CICS files open for output against this data set must be closed before a data set manager, such as DFSMShsm™ or DFSMSdss,[2] can take a backup copy.

If the data set is opened in RLS mode, you need to look in the VSAM catalog to find out whether the data set is eligible for BWO.

UNDETERMINED
Returned for base files if RECOVSTATUS is UNDETERMINED.

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.

Blanks are returned if the access method is BDAM, or if the data set has not been opened by the CICS region in which the command is issued.

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, as a half-word binary value, the numeric journal identifier of the journal being used as the forward-recovery log, if this is a forward-recoverable data set.

FWDRECOVLOG is undefined if the data set is not forward-recoverable. A data set can be defined as being forward recoverable in the ICF catalog or, if it is accessed in non-RLS mode, in the file definition.

---

[2] Hierarchical storage manager (DFSMShsm) and data set services (DFSMSdss) are components of Data Facility Storage Management Subsystem (DFSMS/MVS).

This option is valid for data sets accessed only in non-RLS mode, and for which the recovery attributes are obtained from the file resource definition.

CICS returns a value of zero for forward-recoverable data sets accessed in RLS mode, or for non-RLS mode data sets for which CICS obtains the recovery attribute from the ICF catalog.

FWDRECOVLSN(*data-area***) (VSAM only)**
returns the name (up to 26 characters) of the log stream that is used to log the updates if this is a data set defined with forward-recovery attributes. CICS returns blanks if the data set is not forward recoverable.

The log stream name returned is either:

- The log stream name specified directly in the ICF catalog (for DFSMS/MVS® 1.3 or above), or
- For a non-RLS access mode data set that does not have forward recovery attributes in the ICF catalog, it is a log stream name identified by CICS through a journal name generated from the FWDRECOVLOG value.

LOSTLOCKS(*cvda***) (RLS only)**
returns a CVDA value indicating whether there are any lost locks for this data set. CVDA values are:

NOTAPPLIC
This is not an RLS data set, or the data set has not been opened by the CICS region in which the command is issued.

NOLOSTLOCKS
The data set has no lost locks.

REMLOSTLOCKS
The data set has lost locks, hence is unavailable, but no recovery is required on this CICS region.

RECOVERLOCKS
The data set has lost locks, hence is unavailable, and the CICS region is performing lost-locks recovery.

See the RESETLOCKS and the FORCE|COMMIT|BACKOUT options on the EXEC CICS SET DSNAME command for information about purging UOWs that might be holding up lost locks recovery.

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.

NOTAPPLIC
The data set has not been opened by the CICS region in which the command is issued, or it is a BDAM data set.

PATH
This is a path.

QUIESCESTATE(*cvda***) (VSAM only)**
returns a CVDA value indicating the RLS quiesce state of the data set. The information is obtained from the ICF catalog entry for the data set.

Note: This option is returned, whether or not the data set has been opened by the CICS region in which the command is issued.

CVDA values are:

NOTAPPLIC
This data set is:

- Migrated
- Accessed using BDAM
- Accessed using a level of VSAM that does not support RLS (that is, DFSMS/MVS is earlier than 1.3)

NOTAPPLIC is also returned if CICS is running without RLS support (the RLS=NO system initialization parameter is specified or implied).

QUIESCED
This data set has been quiesced. CICS cannot open files in RLS mode against the data set, and no CICS region has a file currently open against this data set. However, the data set can be opened in non-RLS mode.

QUIESCING
This data set is in the process of quiescing. It applies only to the CICS region that initiated the quiesce; for other CICS regions, UNQUIESCED is returned.

UNQUIESCED
The normal value for a data set that is not quiescing or is not quiesced. It indicates that files can be opened in RLS or non-RLS mode against the data set, the mode being established by the first open. After a file is opened in one mode, other files can be opened only in the same mode.

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, or the data set has not been opened by the CICS region in which the command is issued.

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 or EXEC CICS SET DSNAME REMOVE command, or by an initial or cold start.

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.

RETLOCKS(*cvda***)**
returns a CVDA value indicating whether there are any retained record locks, as a result of deferred recovery work by this CICS region, for the specified data set. CVDA values are:

NOTAPPLIC
This data set has not been opened by the CICS region in which the command is issued.

NORETAINED
This CICS region:

- Has no deferred recovery work for the base data set, and therefore no retained locks, or
- Has recovery work currently in progress.

Note that retained locks may be held against the data set by other CICS regions. The command needs to be issued on all regions in the sysplex to get a full picture of the state of the data set. See the *CICS Recovery and Restart Guide* for information about the CICS batch-enabling sample programs that assist you in doing this, and about the AMS SHCDS LIST subcommands that allow you to investigate retained locks held by CICS regions that are down.

RETAINED
This CICS region has deferred recovery work causing retained locks for the data set. One effect of this is that, if the data set was last opened in RLS mode, the locks are RLS locks and, therefore, the data set cannot be opened in non-RLS mode.

Another effect is that any FILE definitions that specify this data set cannot be changed to specify a different data set.

If the data set is a BDAM data set, or a VSAM data set accessed in non-RLS mode, the locks are CICS record locks, otherwise they are RLS record locks. The UOW that has retained locks is usually shunted, but it may be in the process of being retried.

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.

IOERR
RESP2 values:

40 QUIESCESTATE was specified, but an error was raised by DFSMS/MVS when reading the ICF catalog.
49 An error was raised by DFSMS/MVS when reading the ICF catalog to establish the base data set name.

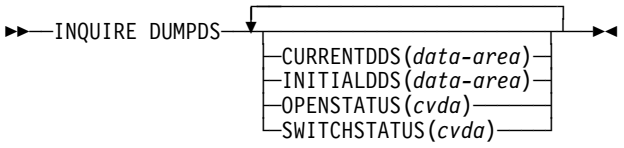
Note: If an IOERR occurs within a browse it does not terminate the browse operation, and CICS attempts to return as many parameter values as possible.

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.



Conditions: NOTAUTH

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.

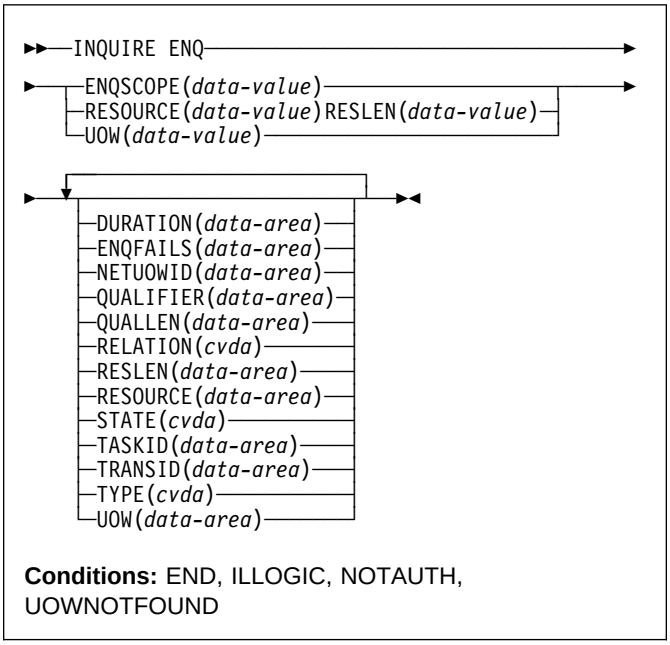
INQUIRE ENQ

Retrieve information about enqueues held or waited on by a UOW, or about UOWs holding or waiting on a specified enqueue.

INQUIRE ENQ is a synonym for INQUIRE UOWENQ; see "INQUIRE UOWENQ" on page 222 for a full description.

Browse only function

The INQUIRE ENQ command, (and the INQUIRE UOWENQ command), can be used only in browse mode.

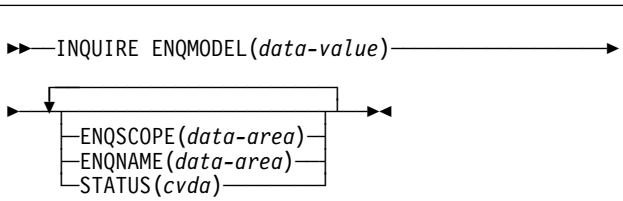


Conditions: END, ILLOGIC, NOTAUTH, UOWNOTFOUND

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

INQUIRE ENQMODEL

Retrieve information about enqueue model definitions on the local system.



Conditions: END, ILLOGIC, NOTAUTH, NOTFND

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

Description

The INQUIRE ENQMODEL command returns information about enqueue model definitions on the local system.

You can make an explicit INQUIRE for a given ENQMODEL, or use the browse form of the command. Browse returns all enqueue model definitions on the local system.

Browsing

To browse through all of the ENQ models in your local system, use the browse options (START, NEXT, and END) on INQUIRE ENQMODEL commands.

See "Browsing resource definitions" on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

ENQMODEL(*data-value***)**
specifies the 8-character identifier of an enqueue model.

ENQSCOPE(*data-area***)**
returns the 4-character name which qualifies sysplex-wide ENQUEUE requests issued by this CICS region. Four blanks indicate that the enqueue is LOCAL.

ENQNAME(*data-area***)**
returns the 1 to 255-character resource name or generic name.

ENQ commands issued by this CICS region are checked against this resource or generic name. If a match is found, and ENQSCOPE was specified, the enqueue will be sysplex-wide, qualified by the 4-character ENQSCOPE.

STATUS(*cvda***)**
returns a CVDA value describing the current state of the ENQMODEL. CVDA values are:

ENABLED
matching enqueue requests are being processed in the normal way.

DISABLED
matching enqueue requests are being rejected, and the issuing tasks are abending with code ANQE. Matching INSTALL CREATE or DISCARD requests are being processed.

WAITING
Matching enqueue requests are being rejected, and the issuing tasks are abending with code ANQE. There are INSTALL CREATE or DISCARD requests waiting to be processed.

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:

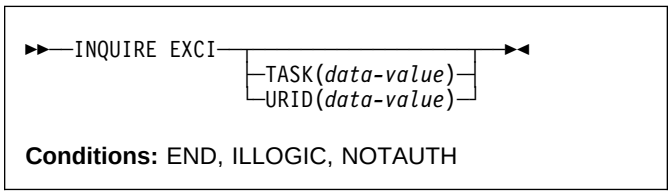
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

NOTFND
RESP2 values:

1 The ENQMODEL cannot be found.

INQUIRE EXCI

Retrieve information about jobs using the external CICS interface.



Conditions: END, ILLOGIC, NOTAUTH

Description

The INQUIRE EXCI command identifies the names of batch jobs currently connected to CICS through the interregion communication (IRC) facility.

Options

TASK(*data-value***)**
specifies, the fullword binary task number of the mirror transaction running on behalf of a specific batch job.

Information about jobs using the external CICS interface is available only after that job has issued at least one DPL request. A nonzero task number indicates that a DPL request is currently active. A zero task number indicates that an external CICS interface session is still open (connected) for that job, although no DPL request is currently active.

URID(*data-value***)**
specifies, when the job is using RRMS to coordinate updates, and when there is an active DPL request for the session, a 32-character string containing the hexadecimal representation of the RRMS Unit of Recovery Identifier.

Conditions

END
RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC
RESP2 values:

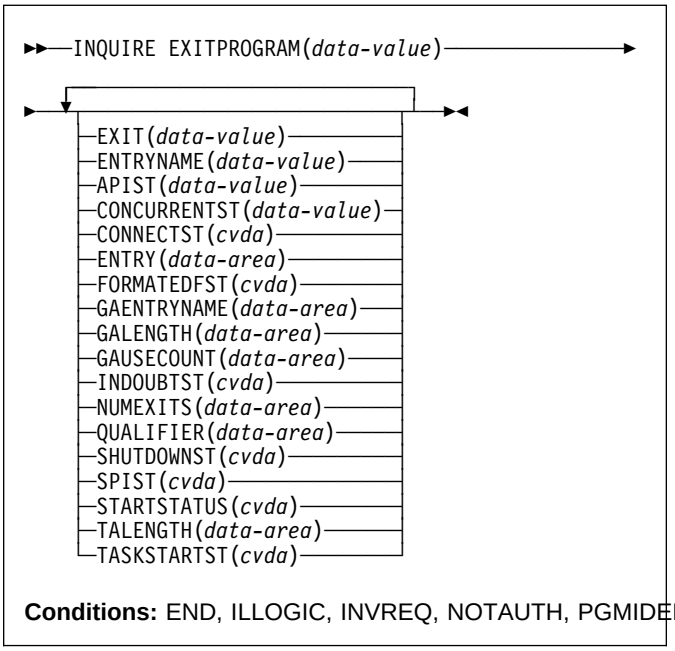
1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

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.

INQUIRE EXITPROGRAM

Retrieve information about a user exit.



Conditions: END, ILLOGIC, INVREQ, NOTAUTH, PGMIDERR

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 is 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

APIST(task-related user exits only)
returns a CVDA indicating which APIs the task-related user exit program uses.

CVDA values are:

BASEAPI
The task-related user exit program is enabled as either QUASIRENT or THREADSAFE, but without the OPENAPI option. This means it is restricted to the CICS permitted programming interfaces.

OPENAPI
The task-related user exit program is enabled with the OPENAPI option. This means it will be permitted to use non-CICS API, for which purpose CICS will give control to the task-related user exit under an L8 mode open TCB. OPENAPI assumes that the program is written to threadsafe standards.

CONCURRENTST
returns a CVDA indicating the concurrency status of the task-related user exit program, as specified by the latest ENABLE command for this program.

CVDA values are:

QUASIRENT
The task-related user exit program is defined as being quasi-reentrant, and is able to run only under the CICS QR TCB when invoking CICS services through the CICS API. To use any MVS services, this task-related user exit program must switch to a privately-managed TCB.

THREADSAFE
The program is defined as threadsafe, and is capable of running under an open TCB. If the APIST option returns OPENAPI, it will always be invoked under an open TCB. If the APIST option returns BASEAPI, it is invoked under whichever TCB is in use by its user task when the program is given control, which could be either an L8 mode open TCB or the CICS QR TCB.

CONNECTST(*cvda***) (task-related user exits only)**
returns a CVDA value indicating the state of the connection between the exit and the external resource manager that it supports. CONNECTST 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.

For example, 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, or DSNCCMD.

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 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.

UNKNOWN
The task-related user exit has been enabled and started, but not enabled for SPI requests. UNKNOWN can also be returned if CICS is unable to call the task related user exit. In both of these cases, CICS cannot tell whether it is connected to its external resource manager.

UNKNOWN will be returned for all subsequent calls for the remaining lifetime of the task. A new task will be able to call the task-related user exit and get the required information.

If the task-related user exit is not enabled, the INQUIRE command returns PGMIDERR. This also indicates that CICS is not connected to the resource manager.

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.

INDOUBTST(*cvda*)
returns a CVDA value indicating whether the task-related user exit is enabled with the INDOUBTWAIT keyword. CVDA values are:

NOTAPPLIC
The exit being inquired upon is a global user exit.

NOWAIT
The exit is not enabled with the INDOUBTWAIT keyword.

WAIT
The exit is enabled with the INDOUBTWAIT keyword.

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.

QUALIFIER(*data-area*)
returns, for a task-related user exit that is enabled for SPI calls, the 8-character qualifier returned by the exit.

For global user exits and task-related user exits that are not enabled for SPI calls, returns blanks.

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.

SPIST(*cvda*)
returns a CVDA value indicating whether the task-related user exit is enabled for SPI calls. CVDA values are:

NOSPI
The exit is not enabled for SPI.

NOTAPPLIC
The exit being inquired upon is a global user exit. This occurs only when the INQUIRE command is explicitly for a global user exit. For example:

```
INQUIRE EXITPROGRAM(abcd) exit(XFCREQ)
```

If you omit EXIT(XFCREQ), you are inquiring about a task-related user exit. Because all global user exits are, by default, task-related user exits as well, NOSPI is returned.

SPI The exit is enabled for SPI.

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:

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.

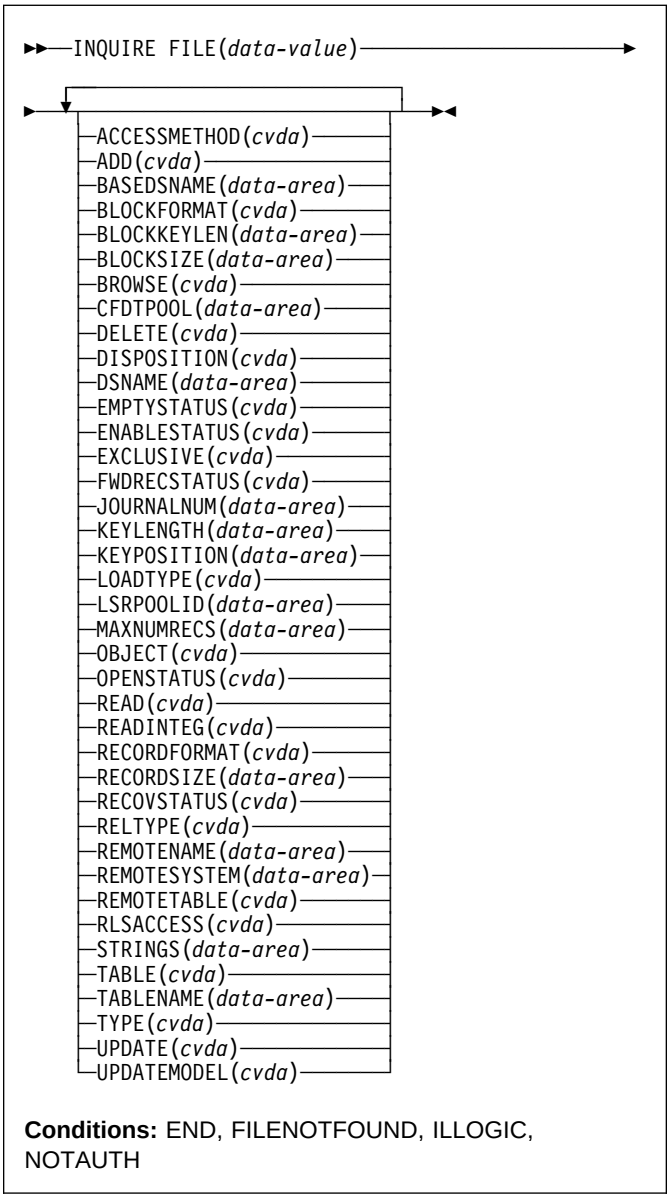
PGMIDERR

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.



Conditions: END, FILENOTFOUND, ILLOGIC, NOTAUTH

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, JOURNALNAME, and TDQUEUE commands if you need information about dump data sets, journals, or TD queues.

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.
- 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 “Null values” on page 13.
- If a file is empty (in VSAM load mode), then, 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 issued against the file before this next access occurs, the file shows CLOSED,ENABLED. This can be a temporary state for a file that has just completed load mode.

Some options for the INQUIRE FILE command are specific to one or another of the file objects supported by CICS, such as VSAM or BDAM data sets, and data tables. Many of these parameters can be specified even when the file refers to a different object from that to which the parameters apply. This is intended to make it easier to switch file definitions between different objects; for example, between non-RLS and RLS access, or between a user-maintained data table and a coupling facility data table. When a parameter is specified for an object to which the file does not currently refer, it is ignored.

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 18 for general information about browsing, including 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. Access to a data table (except while it is being loaded or, for a CICS-maintained data table, when the source data set is being updated or searched for a record that is not in the table), is through CICS data table services. Because this access is still based on VSAM keys, CICS returns VSAM as the access method for any kind of data table.

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.

The BASEDSNAME is blank if the file has not been opened since the last initial or cold start of this CICS. If the file has been opened at least once since the last initial or cold start, CICS returns the 44-character name, even though the file may not be open at the time the command is issued. This is because the name is preserved in the CICS catalog and recovered on a restart.

If the object is a coupling facility data table loaded from a source data set, the 44-character name returned on BASEDSNAME is the same as that returned on DSNAME. BASEDSNAME is blank for a coupling facility data table that is not associated with a source data set.

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.

CFDTPOOL(*data-area***) (CFDT only)**
returns the 8-character name of the coupling facility data table pool in which the coupling facility data table resides. CICS returns blanks if the file does not refer to a coupling facility data table and no pool name has been specified.

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.

If the file has not been opened since the last initial or cold start, the name is taken from the file resource definition. CICS returns blanks if the data set name is not defined on the file definition.

For a coupling facility data table loaded from a data set, CICS returns the 44-character source data set name, and blanks for a coupling facility data table that is not loaded from a data set.

EMPTYSTATUS(*cvda***) (VSAM only)**
returns a CVDA value indicating whether EMPTYREQ has been set for the file. 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, and defined to be used in non-RLS mode, 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 a 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 a 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.

The value CICS returns for FWDRECSTATUS depends on whether the file has been opened since the last initial or cold start:

- If the file has not been opened since the last initial or cold start, CICS returns the value from the file definition.
- If the file has been opened at least once since the last initial or cold start, CICS returns the value that was used when the file was last opened. This can be different from the value on the file definition because, for example, the file definition may be overridden by a value from the ICF catalog.

CVDA values are:

FWDRECOVABLE
The file is forward-recoverable. The RECOVERY option of the FILE definition specifies that updates to the file are to be recorded, to make forward recovery of the file possible. The forward-recovery log can be found using INQUIRE DSNAME.

NOTFWDRCVBLE
The file is not forward-recoverable. CICS returns NOTFWDRCVBLE for a coupling facility data table and a user-maintained data table.

JOURNALNUM(*data-area***)**
returns a halfword binary field indicating the number of the journal to which CICS writes the information required for autojournaling. The value returned in JOURNALNUM is the number specified by the JOURNAL parameter in the file resource definition.

Journal numbers are between 1 and 99 and correspond to journal names DFHJ01 through DFHJ99. A value of 0 means that JOURNAL(NO) is specified and CICS does not perform autojournaling for the file.

JOURNALNUM is ignored for user-maintained and coupling facility data tables: There is no autojournalling of requests made to these tables.

KEYLENGTH(*data-area***)**
returns a fullword binary field indicating the length of the record key for a file associated with a VSAM KSDS or a file associated with a coupling facility data table. If the file is associated with a BDAM data set, the value is the length of the logical key used for deblocking.

Notes:

1. If the file is closed and the key length is not defined in the file definition, the value returned is 0 (zero).
2. If the file is closed and a key length is defined on the file definition, CICS returns the value from the file definition.
3. If the file is open, most files get their key length

from the associated data set, in which case CICS returns the value from the data set. However, files that refer to coupling facility data tables defined with LOAD(NO) must get their keylength from the file definition, in which case CICS returns the value from the file definitions for such files. This value must also match that of the coupling facility data table if it has already been created.

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, or if the file is not open, CICS returns a value of zero for the key position.

For a coupling facility data table associated with a source data set, where the file is open, the key position is obtained from the source data set. If the coupling facility data table is not associated with a source data set CICS returns zero.

LOADTYPE(*cvda***) (VSAM only)**
returns a CVDA value indicating the load type for a coupling facility data table. CVDA values are:

LOAD
The coupling facility data table is, or is to be, preloaded from a source data set.

NOLOAD
The coupling facility data table is not preloaded from a source data set.

NOTAPPLIC
The file is not defined as a coupling facility data table, and no value is defined in the file resource definition.

CICS returns LOAD or NOLOAD if the file is not defined as a coupling facility data table, but one of these options is specified on the LOAD attribute of the file resource definition. In this case, the LOADTYPE CVDA indicates the load type that applies if the file definition is altered to specify TABLE(CF).

LOADTYPE has no significance for a CICS-maintained or user-maintained shared data table. A shared data table is always loaded from a source data set when the first file to reference the table is opened.

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***) (data tables only)**
returns a fullword binary field indicating the maximum number of records that the data table for this file can hold. The value returned by CICS is affected by the following factors:

- If the file resource definition specifies a MAXNUMRECS numeric value, even though the object is not a table (NOTTABLE CVDA is returned on the TABLE option), CICS returns the specified value.
- If the file resource definition is specified with MAXNUMRECS(NOLIMIT), meaning the number of records is unlimited, CICS returns a value of zero. (Internally, CICS holds NOLIMIT as the maximum positive fullword value (+2147483647 or X'7FFFFFFF').
- If the file is remote, CICS returns a value of minus 1 (-1).
- If the object is a coupling facility data table:
  - The maximum number of records can be altered by a coupling facility data table server command, leaving the file definition MAXNUMRECS value unchanged. CICS returns the value in the file definition until the file is opened, after which CICS returns the actual MAXNUMRECS value defined to the server.
  - If the value is changed again by a coupling facility data table server command, CICS obtains and returns the new value only after the file is next opened or inquired upon. Until then, CICS continues to return the old value.
  - You can use the server DISPLAY TABLE console command to obtain the current value for a coupling facility data table.

OBJECT(*cvda***) (VSAM only)**
returns a CVDA value indicating whether the file is associated with a data set (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 data set that is a VSAM base. CICS also returns BASE for data tables (data table access provides primary key access only, not access though a path).

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. Also, if the file has not been opened since the last initial or cold start, CICS returns a default 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.

READINTEG(*cvda***)**
returns a CVDA value indicating the default level of read integrity that is active for the file if a read integrity option is not explicitly coded on a file read request command. CVDA values are:

CONSISTENT
Read requests for this file are subject to consistent read integrity (unless otherwise specified on the read request).

NOTAPPLIC
Read integrity is not applicable for this file for one of the following reasons:

- The file is a VSAM file accessed in non-RLS mode
- The file is a remote file
- The file refers to a BDAM data set
- The file refers to a coupling facility data table.

Note: If you switch a file from RLS to non-RLS mode, the read integrity option specified for RLS mode is preserved. In this case, CICS returns NOTAPPLIC. If you switch the file back to RLS mode, CICS returns the saved read integrity in response to an INQUIRE FILE command.

REPEATABLE
Read requests for this file are subject to repeatable read integrity (unless otherwise specified on the read request).

UNCOMMITTED
No read integrity is specified for this 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 user-maintained 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, or the maximum size of variable-length records.

If the file is not open, CICS returns the value specified in the installed file definition.

If the file is open, most files get their record size from the associated data set, in which case CICS returns the value from the data set. However, files that refer to coupling facility data tables defined with LOAD(NO) must get their record size from the file definition, in which case CICS returns the value from the file definitions for such files. This value must also match that of the coupling facility data table if it has already been created.

RECOVSTATUS(*cvda***)**
returns a CVDA value identifying whether the file is recoverable.

The value CICS returns for RECOVSTATUS depends on whether the file has been opened since the last initial or cold start of the CICS region:

- If the file has not been opened since the last initial or cold start, CICS returns the value from the file definition.
- If the file has been opened at least once since the last initial or cold start, CICS returns the value that was used when the file was last opened. This can be different from the value on the file definition because, for example, the file definition may be overridden by a value from the ICF catalog (but any

value from the ICF catalog is ignored for a user-maintained or CICS-maintained data table)..

CVDA values are:

NOTRECOVABLE
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 (MBBCCHHR) 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.

REMOTETABLE(*cvda*) (VSAM only)
returns a CVDA value indicating whether the file represents an open remote data table. CVDA value can be:

REMTABLE
The file represents an open remote data table.

RLSACCESS(*cvda*)
returns a CVDA value indicating whether the file is defined to be opened in RLS mode. CVDA values are:

NOTAPPLIC
The file is not eligible to be accessed in RLS mode because:

- It is a remote file, or
- It refers to a BDAM data set.

NOTRLS
The file refers to a data set defined to be accessed in non-RLS mode.

RLS The file refers to a data set defined to be accessed in RLS mode.

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.

TABLE(*cvda*) (VSAM and CFDT only)
returns a CVDA value indicating whether the file represents a data table. CVDA values are:

CFTABLE
The file represents a coupling facility data table.

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.

TABLENAME(*data-area*) (CFDT only)
returns the 8-character table name specified for the coupling facility data table on the file resource definition, if one is specified, or returns the file name if the table name is omitted from the file resource definition.

CICS returns blanks if the file does not refer to a coupling facility data table

TYPE(*cvda*)
returns a CVDA value identifying the type of data set that corresponds to this file. The data set must be open to return the type of data set. 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 or the file refers to a data table.

NOTKEYED
The data set is not addressed by physical keys.

RRDS
The data set is a relative record data set.

VRRDS
The data set is a variable—length relative record data set.

NOTAPPLIC
The data set is not open.

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.

| **UPDATEMODEL(***cvda***) (CFDT only)**
| returns a CVDA value indicating the update model specified for the coupling facility data table in the installed file definition. CVDA values are:

| CONTENTION
| The coupling facility data table is updated using the contention model.

| LOCKING
| The coupling facility data table is updated using the locking model.

| NOTAPPLIC
| The file does not refer to a coupling facility data table and UPDATEMODEL on the file resource definition does not specify a value.

| It is possible to define a file that specifies LOCKING or CONTENTION on the UPDATEMODEL attribute when the file does not refer to a coupling facility data table. In this case, CICS returns the specified UPDATEMODEL value on the INQUIRE FILE command, and not NOTAPPLIC. If you redefine the command to refer to a coupling facility data table, the specified UPDATEMODEL takes effect.

| For information about the contention and locking models, see the *CICS Resource Definition Guide*.

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:

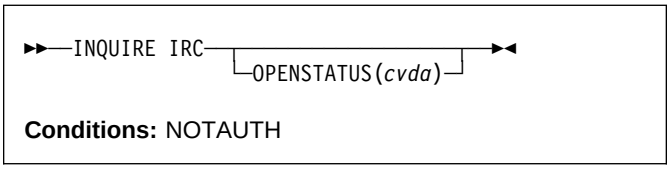
1 You have issued a START command when a browse of this resource type is already in progress, or you have issued a NEXT or an END command when a browse of this resource type is not in progress.

NOTAUTH
RESP2 values:

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 IRC

Show the IRC status.



Conditions: NOTAUTH

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.

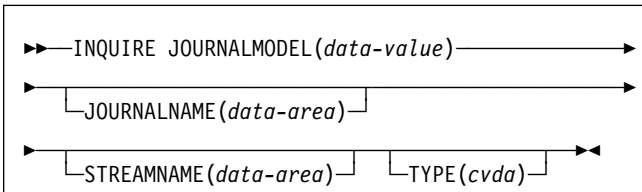
Conditions

NOTAUTH
RESP2 values:

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

INQUIRE JOURNALMODEL

Retrieve information about installed journal models, thus enabling you to obtain corresponding log stream names.



Conditions: END, ILLOGIC, NOTAUTH, NOTFND

Description

The INQUIRE JOURNALMODEL command returns information about a particular installed journal model and enables you to obtain corresponding log stream names.

Browsing

You can also browse through all of the journal model names on your system by using the browse options (START, NEXT, and END) on INQUIRE JOURNALMODEL commands. See "Browsing resource definitions" on page 18 for general information about browsing, including syntax, exception conditions, and examples.

CICS returns journal models in alphanumeric sequence of the JOURNALNAMEs specified in the journal model, but with specific names being returned before the generic names. The following examples of journal names defined on journal models show the order in which the journal models are returned on a browse JOURNALMODEL operation:

DFHJ15
DFHJ25
DFHJ%0
DFH*
USERJNL1
USERJNL2
USERJNL*

Options

JOURNALMODEL(*data-value***)**
specifies the 1- to 8-character name of an installed journal model.

JOURNALNAME(*data-area***)**
returns the 1- to 8-character journal name, which can be a specific or a generic name. See the *CICS Resource Definition Guide* for further information about the JOURNALNAME operand.

STREAMNAME(*data-area*)
returns the MVS log stream name (LSN) associated with the JOURNALMODEL entry.

The name can be a specific LSN, or a template using a maximum of any 3 of the 4 symbols symbols &USERID, &APPLID, &JNAME and &SYSID.

The name, LSN or template, can be up to 26 characters in length. Names less than 26 character are padded with trailing blanks (X'40').

TYPE(*cvda*)
indicates the log stream type. The CVDA values are:

DUMMY	Records are not written to any log stream.
MVS	Records are written to an MVS log stream.
SMF	Records are written to the MVS SMF log stream.

Conditions

END
RESP2 values:

2 All authorized resources have been retrieved. All data areas specified on this command are left unchanged.

ILLOGIC
RESP2 values:

1 A START has been given when a browse is already in progress, or a NEXT or END has been given without a preceding START.
2 The browse token is not valid.

NOTAUTH
RESP2 values:

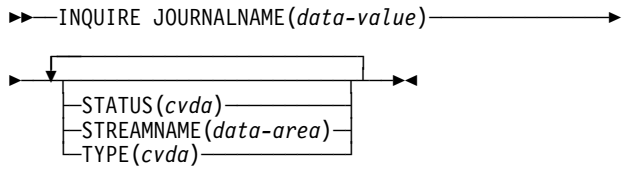
100 The user is not authorized for this command.

NOTFND
RESP2 values:

1 The specified journal model was not found.

INQUIRE JOURNALNAME

Retrieve information about the status of the system log and general logs.



Conditions: END, ILLOGIC, JIDERR, NOTAUTH

Description

The INQUIRE JOURNALNAME command returns information about the journals (including the system log and general logs) on your system.

Browsing

You can also browse through all the journal entries in the journal names table on your system by using the browse options (START, NEXT, and END) on INQUIRE JOURNALNAME commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

JOURNALNAME(*data-value*)
specifies a 1- to 8-character journal name.

To inquire on journals defined with a numeric identifier in the range 1–99, specify journal name DFHJ*nn*, where *nn* is the journal number.

To inquire on the system log, specify DFHLOG.

STATUS(*cvda*)
indicates the status of the journal. CVDA values are:

DISABLED
The journal has been disabled by a CEMT, or EXEC CICS, SET JOURNALNAME(...) command. It cannot be used until it is reenabled by the STATUS(ENABLED) or ACTION(RESET) options on a SET JOURNALNAME command.

ENABLED
The journal is installed and is available for use.

FAILED
The journal has experienced a log stream failure. It cannot be used until it is reenabled by the STATUS(ENABLED) or ACTION(RESET) options

on a SET JOURNALNAME command, or until after the next CICS restart. The log stream should be deleted from the MVS system logger inventory before being used again.

STREAMNAME(*data-area***)**
returns the MVS logger log stream name (LSN) associated with the journal name.

The name can be up to 26 characters in length. Names less than 26 character are padded with trailing blanks (X'40'). If the journal is defined by a journal model that specifies a type of DUMMY or SMF, CICS returns 26 blanks.

TYPE(*cvda***)**
Indicates the type of log stream format. CVDA values are:

DUMMY Records are not written to any log stream.

MVS Records are written to an MVS logger log stream.

SMF Records are written to the MVS SMF log stream.

Conditions

END
RESP2 values:

2 All authorized resource definitions have been retrieved. All data areas specified on this command are left unchanged.

ILLOGIC
RESP2 values:

1 A START has been given when a browse is already in progress, or a NEXT, or an END, has been given without a preceding START.

JIDERR
RESP2 values:

1 The specified journal name was not found.

NOTAUTH
RESP2 values:

100 The user is not authorized for this command.
101 The user does not have the required access to the specified journal. (Not applicable to INQUIRE JOURNALNAME START, INQUIRE JOURNALNAME NEXT, or INQUIRE JOURNALNAME END commands.)

INQUIRE JOURNALNUM

This command is supported in releases of CICS earlier than CICS Transaction Server for OS/390 for retrieving information about the system log and user journals.

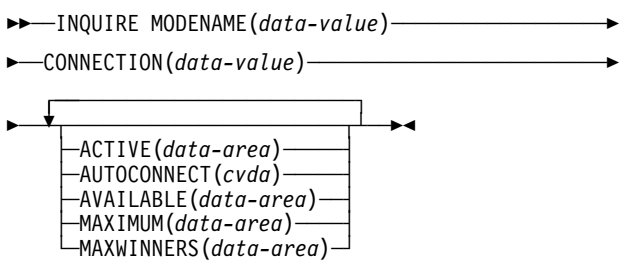
Description

For CICS Transaction Server for OS/390, this command is replaced by the INQUIRE JOURNALNAME command. All the options on INQUIRE JOURNALNUM are obsolete, and the only run-time support provided by CICS for compatibility with earlier releases is to return the JIDERR exception condition. The translator translates the command, but issues a warning message.

The browse function is provided for compatibility with releases of CICS earlier than CICS Transaction Server for OS/390. A NORMAL condition is returned for the START browse and END browse operations. The END condition is returned for the NEXT browse operation.

INQUIRE MODENAME

Retrieve information about a session group within a connection.



Conditions: END, ILLOGIC, NOTAUTH, SYSIDERR

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 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

ACTIVE(*data-area***)**
returns a halfword binary field giving the number of sessions within the group that 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:

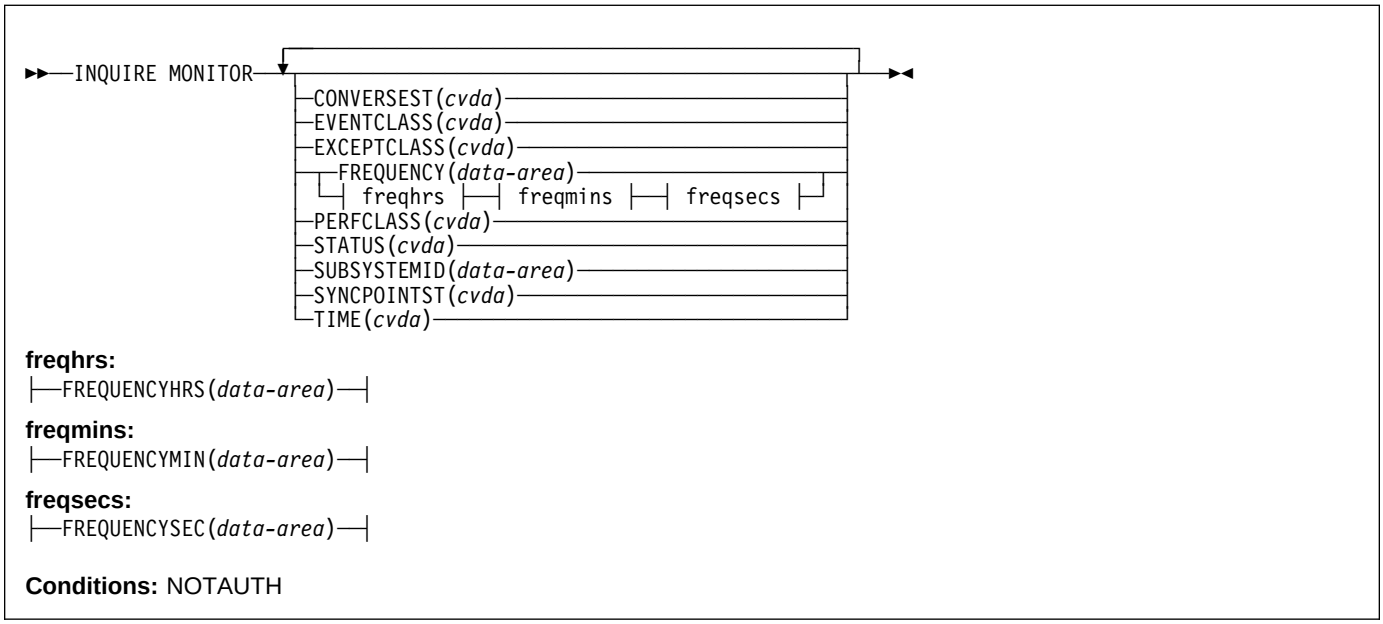
1 The connection cannot be found.

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 278 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(*data-area***)**
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).

FREQUENCYSEC(*data-area***)**
returns the seconds component of the frequency interval, in fullword binary form (see the FREQUENCY option).

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.

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 Performance Guide*. For information on how to specify the subsystem identification, see the MNSUBSYS parameter in the *CICS Resource Definition Guide*.

SYNCPOINTST(*cvda***)**
returns a CVDA value indicating whether CICS records performance class data separately for each unit of work (UOW) within tasks that contain multiple UOWs. A UOW 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 UOW begins immediately after, except at end of task. When rollback occurs on a syncpoint, the UOW does not end. CVDA values are:

NOSYNCPOINT
Performance data is combined over all UOWs in a task for recording.

SYNCPOINT
Performance data is recorded separately for each UOW.

TIME(*cvda***)**
returns a CVDA value identifying whether the performance class time-stamp fields returned to an application using the COLLECT STATISTICS MONITOR command are 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 Customization Guide* for information on the SMF header. CVDA values are:

GMT Time stamps are Greenwich mean time.

LOCAL
Time stamps are local time.

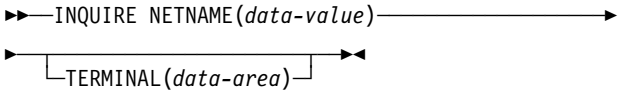
Conditions

NOTAUTH
RESP2 values:

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

INQUIRE NETNAME

Retrieve information about a terminal or session.



Conditions: NOTAUTH, TERMIDERR

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 network identifier, instead of its CICS terminal identifier. For a physical terminal, the network identifier is the name by which the 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.

Thus, compared with INQUIRE TERMINAL, 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.

If there are multiple entries for a netname, and the inquiry is not part of a browse, the first entry found is returned. Entries are searched in the following sequence:

1. VTAM terminals and consoles, in alphanumeric sequence.
2. Connections, in alphanumeric sequence. The leading session is returned (in a browse, all sessions are returned).

Note that this order is not guaranteed to be maintained in future releases.

Browsing

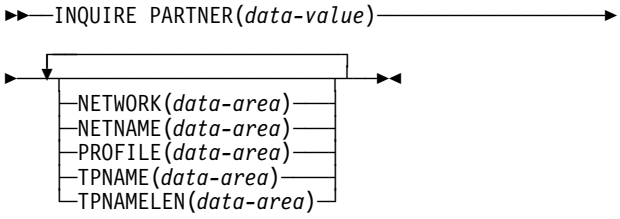
You can also browse through the definitions of all the netnames installed in your system by using the browse options (START, NEXT, and END) on INQUIRE NETNAME or INQUIRE TERMINAL commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Note that connections without sessions, such as indirect connections, or remote connections that are not in use, are not returned. If you want all connections to be returned, use the INQUIRE CONNECTION command.

See “INQUIRE TERMINAL” on page 190 for details of the options and conditions that apply to the INQUIRE NETNAME command.

INQUIRE PARTNER

Retrieve information about a partner.



Conditions: END, ILLOGIC, NOTAUTH, PARTNERIDERR

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 18 for general information about browsing, including 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 can be up to 64 characters long; you can determine the actual length with the TPNAMELEN option.

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.

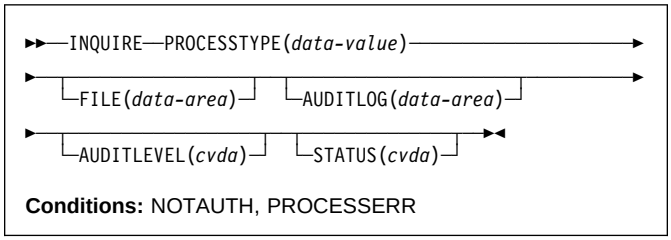
PARTNERIDERR
RESP2 values:

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

INQUIRE PROCESSTYPE

Browsing

Retrieve the attributes of a CICS business transaction services (BTS) process-type.



Conditions: NOTAUTH, PROCESSERR

Description

INQUIRE PROCESSTYPE returns the attributes of a specified process-type.

Browsing

You can also browse through all of the process-type definitions in your system by using the browse options (START, NEXT, and END) on INQUIRE PROCESSTYPE commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

AUDITLEVEL(cvda)
indicates the level of audit currently active for processes of the specified type. CVDA values are:

ACTIVITY
Activity-level auditing. Audit records are written from:

1. The process audit points
2. The activity primary audit points.

FULL
Full auditing. Audit records are written from:

1. The process audit points
2. The activity primary *and* secondary audit points.

OFF
No audit trail records are written.

PROCESS
Process-level auditing. Audit records are written from the process audit points only.

For details of the records that are written from the process, activity primary, and activity secondary audit points, see the *CICS Business Transaction Services* manual.

AUDITLOG(data-area)
returns the 8-character name of the CICS journal used as the audit log for processes of the specified type.

FILE(data-area)
returns the 8-character name of the CICS file associated with the process-type.

PROCESSTYPE(data-value)
specifies the name (1–8 characters) of the process-type being inquired upon.

STATUS(cvda)
indicates whether new processes of the specified type can currently be defined. CVDA values are:

DISABLED The installed definition of the process-type is disabled. New processes of this type cannot be defined.

ENABLED The installed definition of the process-type is enabled. New processes of this type can be defined.

Conditions

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 resource in the way requested.

PROCESSERR
RESP2 values:

1 The process-type specified on the PROCESSTYPE option could not be found.

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 18 for general information about browsing, including 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:

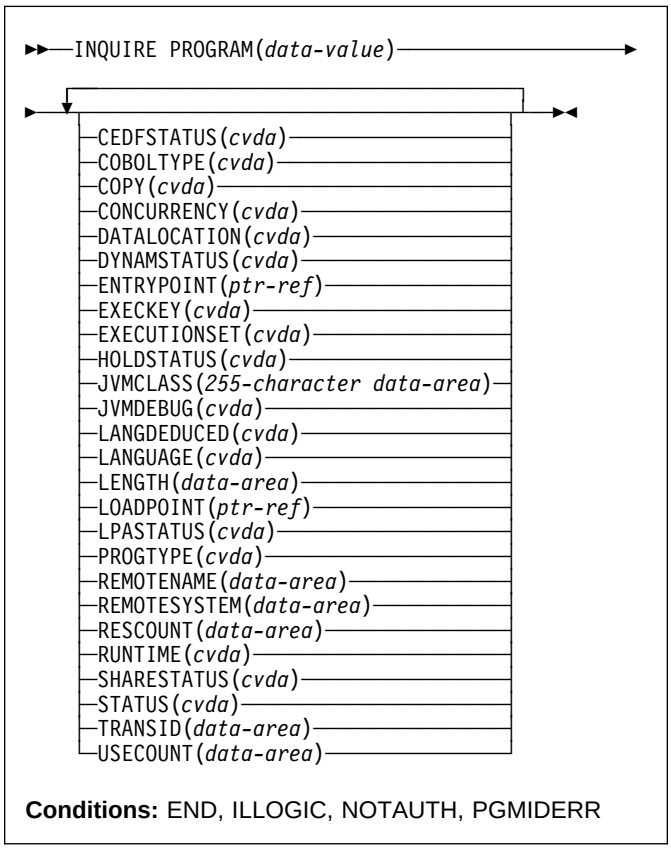
100 The user associated with the issuing task is not authorized to use this command.

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 inspects a module only if it is already loaded and is the copy currently available for use. CICS does not do a load for an INQUIRE PROGRAM command, nor attempt to autoinstall a resource for which it has no definition.

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 18 for general information about browsing, including 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. This CVDA value is always returned for JVM programs.

NOTINIT
: The module is defined as a COBOL program, but the type cannot be determined because the module has not been loaded yet.

CONCURRENCY
returns a CVDA indicating the concurrency attribute of the installed program definition. The CVDA values are:

QUASIRENT
: The program is defined as being quasi-reentrant, and is able to run only under the CICS QR TCB.

THREADSAFE
The program is defined as threadsafe, and is able to run under whichever TCB is in use by its user task when the program is given control. This could be either an open TCB or the CICS QR TCB.

Notes:

1. If the program is not yet loaded (or is waiting to be reloaded following a NEWCOPY or PHASEIN request), the concurrency attribute is derived from the installed program resource definition. Note that the default for the program definition is QUASIRENT. However, in the case of a Language Environment-conforming program, the concurrency as originally defined can be overridden when the program is subsequently loaded. If CICS finds that the program itself contains a CONCURRENCY value defined by LE run-time options, the installed program resource definition is updated by the LE run-time option.
2. The CONCURRENCY attribute on the installed program resource definition is not changed by the FORCEQR system initialization parameter. CICS returns a CVDA of THREADSAFE for a threadsafe-defined program, even if FORCEQR=YES is specified.
3. The CONCURRENCY attribute on the installed program resource definition for a task-related user exit program is not changed by any options specified on an ENABLE command. For a task-related user exit program, CICS always returns a CVDA using the values defined in the program resource definition.

   You cannot modify a program's concurrency attribute using the SPI—the CONCURRENCY option is not supported on the EXEC CICS SET PROGRAM command. You can only change the concurrency by on of the following methods:

   - Redefine the program's CONCURRENCY option in the CICS program resource definition, or in the program autoinstall model, and reinstall the definition
   - Recompile and link-edit the program with new LE run-time options, and issue a CEMT, or EXEC CICS, SET PROGRAM(...) NEWCOPY (or PHASEIN) command.

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. This CVDA value is always returned for JVM programs.

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.

DYNAMSTATUS(*cvda*) (programs only)
returns a CVDA value indicating whether, if the program is the subject of a program-link request, the request can be dynamically routed. CVDA values are:

DYNAMIC
If the program is the subject of a program-link request, the CICS dynamic routing program is invoked. Providing that a remote server region is not named explicitly on the SYSID option of the LINK command, the routing program can route the request to the region on which the program is to execute.

NOTDYNAMIC
If the program is the subject of a program-link request, the dynamic routing program is not invoked.

For a distributed program link (DPL) request, the server region on which the program is to execute must be specified explicitly on the REMOTESYSTEM option of the PROGRAM definition or on the SYSID option of the LINK command; otherwise it defaults to the local region.

For information about the dynamic routing of DPL requests, see the *CICS Intercommunication Guide*.

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, or is a JVM 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 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.

JVMCLASS(*255–character data-area***) (JVM programs only)**
: returns the name of any class specified in the program definition.

JVMDEBUG(*cvda***) (JVM programs only)**
: returns a CVDA value indicating whether or not the JVM is to operate in debugging mode for this program. CVDA values are:

DEBUG
: The JVM is to operate in debugging mode for this program.

NODEBUG
: The JVM is not to run in debugging mode for this program.

LANGDEDUCED(*cvda***) (programs only)**
: returns a CVDA value indicating the language in which the module is written, if this is known. If the module is not yet loaded, CICS cannot deduce the language. In this case, the CVDA value indicates the defined language taken from the resource definition. CVDA values are:

ASSEMBLER
: The language is assembler.

C
: The language is C/370, C++, or VisualAge for Java, Enterprise ToolKit for OS/390 (replace with symbols!!!)

  Note: It is intended in a future release to report C++ (symbol) separately, and for HPJ programs to return a LANGDEDUCED CVDA of JAVA. JAVA and VisualAge for Java, Enterprise ToolKit for OS/390 programs will be distinguished by using the RUNTIME option, which will report JVM for JVM programs and LE370 for VisualAge for Java, Enterprise ToolKit for OS/390 programs.

COBOL
: The language is VS COBOL II or COBOL/370™.

JAVA
: The language is Java. This CVDA is currently returned for JVM programs only.

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, and has not been loaded.

PLI or PL1
: The language is PL/I.

LANGUAGE(*cvda***) (programs only)**
: returns a CVDA value indicating the language with which the module is defined. This value is taken from the 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,or a JVM program.

LOADPOINT(*ptr-ref***)**
returns the load address of the module. If it is not currently loaded, or if the program is running under a JVM , 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, is a remote program, or is a JVM 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 partition 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 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 either a remote program, or a JVM program.

RUNTIME(*cvda***) (JVM programs only)**
returns a CVDA value indicating the runtime environment of the program. CVDA values are:

JVM The program is a Java program that will run under the control of a Java Virtual Machine.

LE370
The program will run with LE370 runtime support.

NONLE370
The program will run with a language-specific runtime environment.

UNKNOWN
The program runtime environment is unknown, because the program has not been loaded by CICS, and therefore its source language has not been deduced, which dictates the runtime environment to be used.

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 or a JVM program.

PRIVATE
The module is loaded from the concatenated libraries named on the DFHRPL DD statement.

SHARED
The LPA copy is to be used, if one is available. If it is not, the module is 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, or a JVM program.

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:

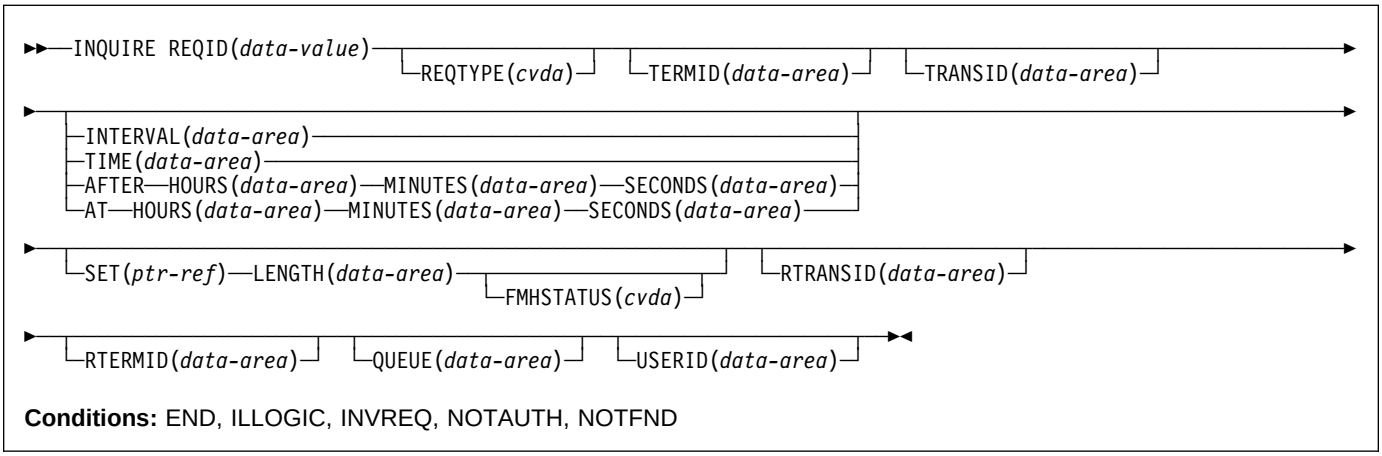
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

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.

INQUIRE REQID

Retrieve information about a queued request.



Conditions: END, ILLOGIC, INVREQ, NOTAUTH, NOTFND

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

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. For example, a request to START a transaction may be delayed beyond expiry time, 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.
- Separate hours, minutes, and seconds, which you obtain by specifying HOURS, MINUTES, and SECONDS with either AT or AFTER.

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 18 for general information about browsing, including syntax, exception conditions, and examples.

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

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:

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:

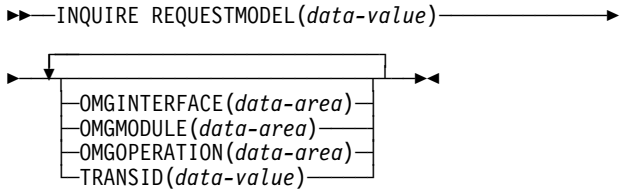
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The REQID cannot be found.

INQUIRE REQUESTMODEL

Retrieve information about a REQUESTMODEL.



Conditions: END, ILLOGIC, JIDERR, NOTAUTH

Description

A REQUESTMODEL resource definition maps an inbound request that is formatted using the Internet Inter-ORB PROTOCOL (IIOP) to a CICS transaction that is to be started to process the request. The INQUIRE REQUESTMODEL command returns information about installed REQUESTMODELs.

Browsing

You can also browse through all the REQUESTMODELs that are installed on your system by using the browse options (START, NEXT, and END) on INQUIRE REQUESTMODEL commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples. You can specify the following options on the INQUIRE request:

Options

REQUESTMODEL(*data-value***)**
specifies an 8-character request model name.

OMGMODULE*(data-area)*
returns the 58-character field containing the pattern defining the name scope of the interface and operation values for this REQUESTMODEL.

OMGINTERFACE*(data-area)*
returns the 31-character field containing the pattern matching the interface name for this REQUESTMODEL.

OMGOPERATION*(data-area)*
returns the 31-character field containing the pattern matching the IDL operation name for this REQUESTMODEL.

TRANSID*(name)*
returns the 4-character name of the CICS transaction to be executed when a request matching the specification of the REQUESTMODEL is received.

Conditions

END
RESP2 values:

2 All authorized resource definitions have been retrieved. All data areas specified on this command are left unchanged.

ILLOGIC
RESP2 values:

1 A START has been given when a browse is already in progress, or a NEXT or an END, has been given without a preceding START.

NOTAUTH
RESP2 values:

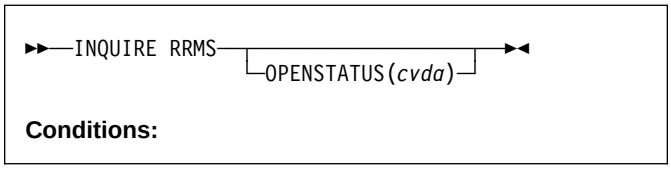
100 The user is not authorized for this command.

NOTFND
RESP2 values:

1 The specified request model was not found.

INQUIRE RRMS

Retrieves the status of transactional EXCI.



Conditions:

Description

The INQUIRE RRMS command indicates whether inbound transactional EXCI work is currently being accepted.

Options

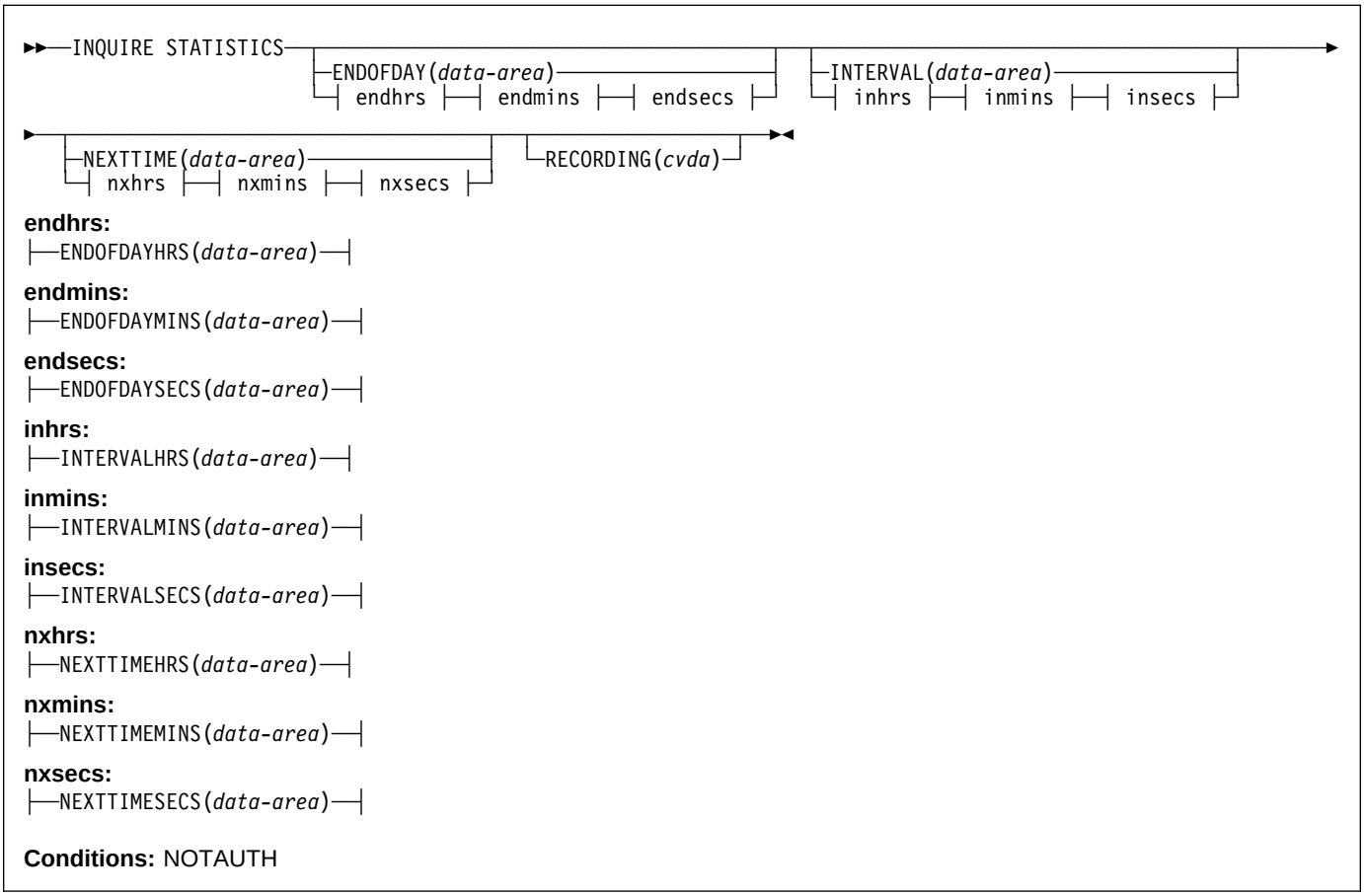
OPENSTATUS(*cvda***)**
returns a CVDA value indicating whether CICS accepts inbound transactional EXCI work or not. CVDA values are:

CVDA_OPEN
indicates that CICS does accept inbound transactional EXCI work.

CVDA_CLOSED
indicates that CICS does not accept inbound transactional EXCI work.

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 Performance Guide* contains more detail about CICS statistics, and the SET STATISTICS command on page 285 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***)**
returns a CVDA value indicating whether the recording of interval statistics is switched on or off. End-of-day, unsolicited, and requested 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 237, or by a CEMT PERFORM STATISTICS transaction.)

CVDA values are:

OFF Recording is off.

ON Recording is on.

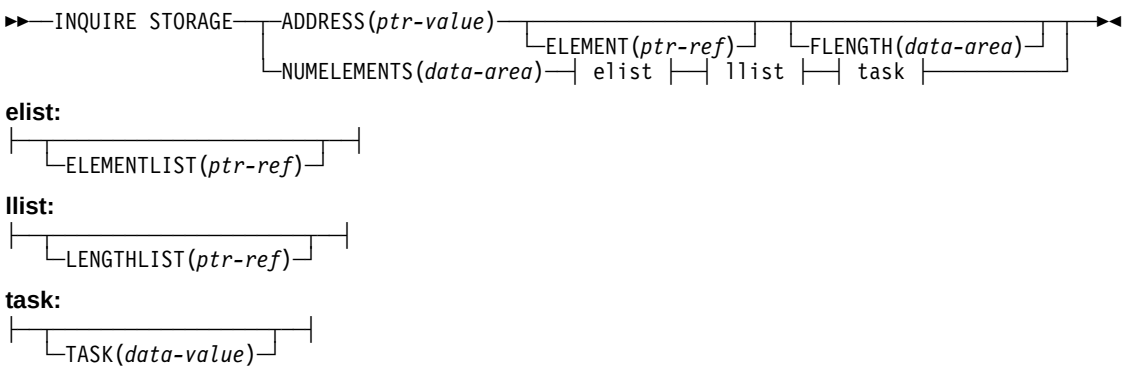
Conditions

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.



Conditions: NOTAUTH, TASKIDERR

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.

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:

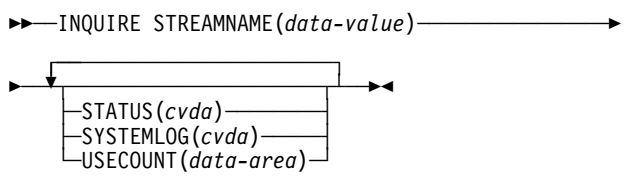
100 The user associated with the issuing task is not authorized to use this command.

TASKIDERR
RESP2 values:

1 The task number does not exist.
2 The task number designates a system task, not a user task.

INQUIRE STREAMNAME

Retrieve information about a currently connected MVS log stream.



Conditions: END, ILLOGIC, NOTAUTH, NOTFND

Description

The INQUIRE STREAMNAME command allows you to look at information about a currently connected MVS log stream.

Browsing

You can also browse through log stream names by using the browse options (START, NEXT, and END) on INQUIRE STREAMNAME commands. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

STATUS(*cvda***)**
returns a CVDA value indicating the status of the log stream. CVDA values are:

FAILED
The message logger has detected a problem with the specified log stream.

OK No errors have been detected.

STREAMNAME(*data-value***)**
specifies an MVS system logger log stream name.

CICS returns a NOTFND condition if the log stream name does not exist, or if there are no longer any users of the log stream in this CICS region (see the USECOUNT option).

SYSTEMLOG(*cvda***)**
returns a CVDA value indicating whether the log stream is the system log. CVDA values are:

NOSYSLOG
The log stream is not the system log.

SYSLOG
The log stream is the system log.

USECOUNT(*data-area***)**
returns the number of CICS journal names and forward recovery logs within this CICS system that are currently using the log stream.

The use count is always at least 1, because CICS does not maintain any information about a log stream that no longer has any users, in which case an INQUIRE STREAMNAME command returns a NOTFND condition.

If the log stream name refers to the CICS system log, the use count is always 1. This is so, even when user application programs write recovery records to the CICS system log.

Conditions

END
RESP2 values:

2 All authorized resource definitions have been retrieved. All data areas specified on this command are left unchanged.

ILLOGIC
RESP2 values:

1 A START has been given when a browse is already in progress or a NEXT or an END has been given without a preceding START.
2 The browse token is not valid.

NOTAUTH
RESP2 values:

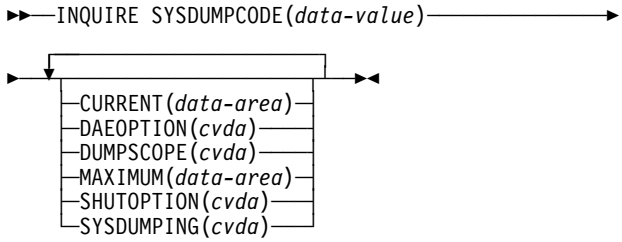
100 The user is not authorized for this command.

NOTFND
RESP2 values:

1 The requested log stream name was not found.

INQUIRE SYSDUMPCODE

Retrieve information about a system dump table entry.



Conditions: END, ILLOGIC, NOTAUTH, NOTFND

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 which 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 are 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 an initial or 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 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 18 for general information about browsing, including 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 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 does 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 *OS/390 MVS Diagnosis: Tools and Service Aids* 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, it is 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 Version 5 or later 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 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 INQUIRE SYSTEM command on page 170). MVS may also be allowed to suppress the dump if appropriate (the DAE option). CVDA values are:

NOSYSDUMP
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:

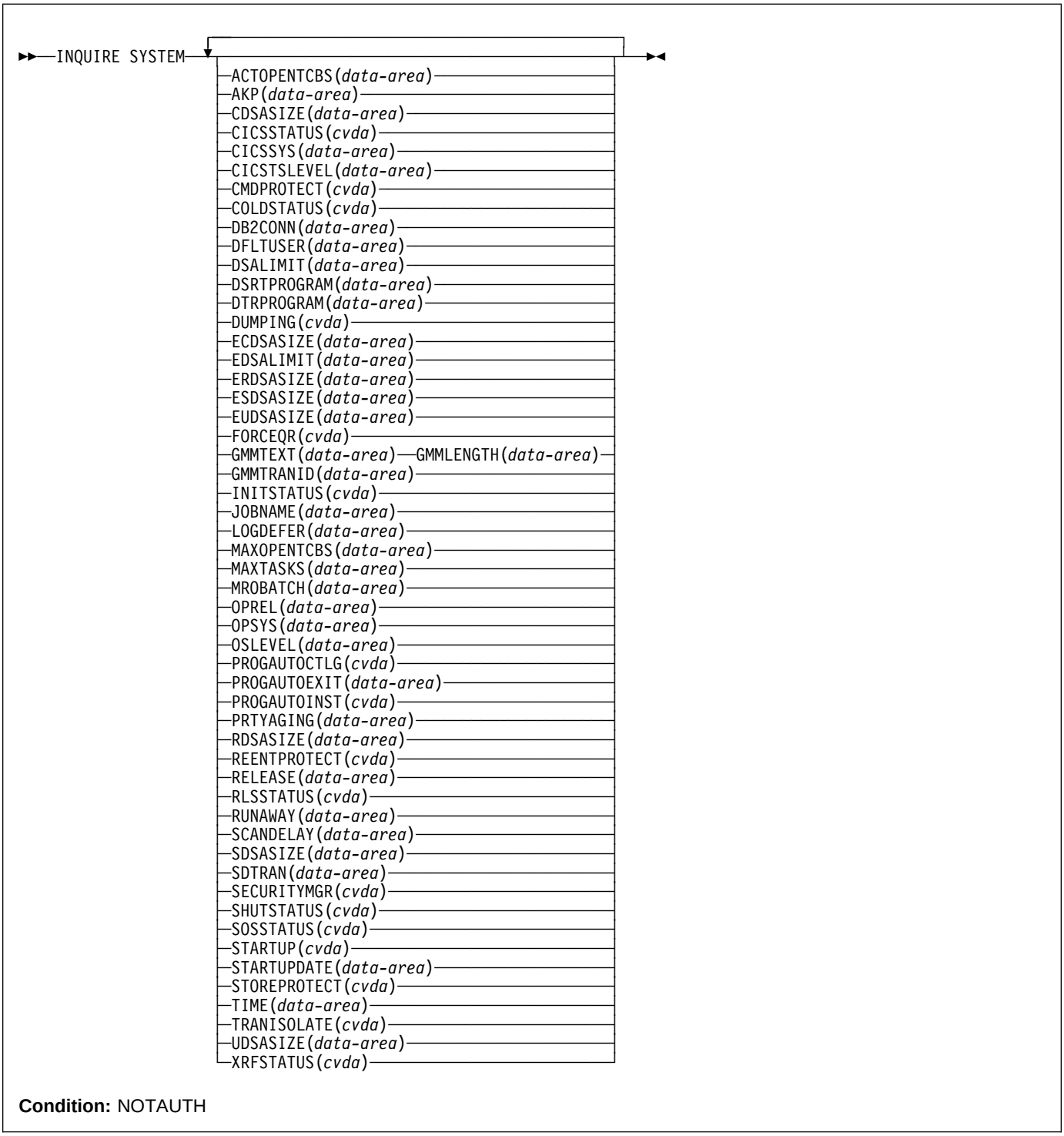
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The named dump code cannot be found.

INQUIRE SYSTEM

Retrieve CICS system information.



Condition: NOTAUTH

For more information about the use of CVDAs, see
“CICS-value data areas (CVDAs)” 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 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 System Definition Guide* is a good source of additional information. For state options, the *CICS Customization Guide* is the primary source.

Table 2. INQUIRE SYSTEM options

Option	Origin
ACTOPENTCBS	System state
AKP	AKPFREQ in SIT
CDSASIZE	System state
CICSSTATUS	System state
CICSSYS	System state
CICSTSLEVEL	CICS control block
CMDPROTECT	CMDPROT in SIT
COLDSTATUS	System state
DB2CONN	DB2CONN in SIT
DFLTUSER	DFLTUSER in SIT
DSALIMIT	DSALIM in SIT
DSRTPROGRAM	DSRTPGM 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
FORCEQR	FORCEQR in SIT
GMMTEXT, GMMLENGTH	GMTEXT in SIT
GMMTRANID	GMTRAN in SIT
INITSTATUS	System state
JOBNAME	JCL or cataloged procedure
LOGDEFER	LGDFINT in SIT
MAXOPENTCBS	MAXOPENTCBS in SIT
MAXTASKS	MXT in SIT
MROBATCH	MROBTCH in SIT
OPREL	Operating system (MVS)
OPSYS	Operating system (MVS)
OSLEVEL	Operating system (OS/390)
PROGAUTOCTLG	PGAICTLG in SIT
PROGAUTOEXIT	PGAIEXIT in SIT
PROGAUTOINST	PGAIPGM in SIT
PRTYAGING	PRTYAGE in SIT
RDSASIZE	System state
REENTPROTECT	RENTPGM in SIT
RELEASE	CICS system code
RLSSTATUS	RLS in SIT
RUNAWAY	ICVR in SIT
SCANDELAY	ICVTSD in SIT
SDSASIZE	System state
SDTRAN	SDTRAN in SIT
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 Transaction Server for OS/390. The translator accepts them and gives a warning. At run time, the data areas provided are left unchanged.

Options

ACTOPENTCBS(*data-area***)**
returns a fullword binary field giving the number of open TCBs currently allocated to user tasks.The open TCBs are allocated from the pool of open TCBs that CICS attaches up to the maximum set by the MAXOPENTCBS system initialization parameter. The ACTOPENTCBS value can be equal to, or less than , the MAXOPENTCBS value. If it is equal to MAXOPENTCBS, tasks that require an open TCB are made to wait.

AKP(*data-area***)**
returns a fullword binary field giving the activity keypoint trigger value, which is the number of write requests to the CICS system log stream output buffer 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 16MB (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 on page 171 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.

CICSTSLEVEL(*data-area***)**
returns a 6-character value identifying the version, release, and modification level of the CICS Transaction Server for OS/390® product under which the CICS region is running. The value is of the form *vvrrmm*, and CICS Transaction Server for OS/390 Release 3 returns 010300.

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.

COLDSTATUS(*cvda***)**
returns a CVDA value indicating whether CICS performed a cold or an initial start. (For information about the types of CICS startup, see the *CICS Recovery and Restart Guide*.)

The CVDA values are:

COLD
CICS performed a cold start. Log information about local resources was erased, but information about the outcome of local units of work, needed to allow remote systems or RMI-connected resource managers to resynchronize their resources, was preserved.

INITIAL
CICS performed an initial start. All log information about both local and remote resources was erased.

NOTAPPLIC
CICS performed neither a cold nor an initial start.

DB2conn(*data-area***)**
returns the 1-8 character name of the DB2CONN currently installed, or blanks if no DB2CONN is currently installed. DB2CONN allows the user to determine the name of the RDO DB2CONN definition. Only one DB2CONN can be installed at a time. A DB2CONN defines the global attributes of the connection to be established between DB2 and CICS.

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.)

DSRTPROGRAM(*data-area***)**
returns the 8-character name of the distributed routing program.

DTRPROGRAM(*data-area***)**
returns the 8-character name of the dynamic routing program.

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).

FORCEQR
returns a CVDA value indicating whether quasi-reentrancy is currently being forced for all user application programs that are defined as threadsafe. The CVDA values are:

FORCE All user application programs are being forced to execute under the QR TCB, as if they were defined with the CONCURRENCY(QUASIRENT) attribute, even if they were defined as threadsafe.

NOFORCE Quasi-reentrancy is not being enforced for all user application programs, and the threadsafe attribute on program resource definitions is nbeing honored.

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 be long enough to accommodate 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 Customization Guide* for more information about CICS initialization.

JOBNAME(*data-area***)**
returns the 8-character MVS jobname under which CICS is running.

LOGDEFER(*data-area***)**
returns the halfword binary value giving the log deferral interval, which is the period of time used by CICS Log Manager when determining how long to delay a forced journal write request before invoking the MVS system logger. See the *CICS System Definition Guide* for information about the LOGDEFER parameter and associated SIT parameter LGDFINT.

MAXOPENTCBS(*data-area***)**
returns a fullword binary field giving the maximum number of open TCBs that CICS is allowed to attach and maintain in its pool of open TCBs. For information about the number actually allocated, see the ACTOPENTCBS option.

The difference between MAXOPENTCBS and ACTOPENTCBS represents the number of open TCBs that are free.

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 Transaction Server for OS/390 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***) (supported for compatibility only)**
returns a halfword binary field giving the last 2 digits of the level number of the operating system under which the CICS region is running. For example, OS/390 Release 4 is represented by 04.

Note: This field is supported for compatibility purposes only. The information is derived from the last two numbers held in the MVS CVTPRODN field. For example, CVTPRODN holds the value SP5.2.2 for MVS/ESA SP™ Version 5 Release 2.2 (in which case OPREL returns 22), and SP6.0.3 for OS/390 Release 3. See the OSLEVEL field for the full version and release number of OS/390.

OPSYS(*data-area***)**
returns a 1-character value identifying the operating system under which CICS is running. A value of "X" represents MVS/ESA.

OSLEVEL(*data-area***)**
returns a 6-byte field containing the version, release, and modification level of the OS/390 product on which CICS is running. For example, OS/390 Version 2 Release 4 Modification 0 returns the value 020400.

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 cataloged 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***) (supported for compatibility only)**
returns a 4-character string comprising the value 0530, indicating the level number of the CICS code within the CICS Transaction Server for OS/390.

This option is supported only for compatibility with earlier releases. As an exclusive element of CICS Transaction Server for OS/390, CICS does not have a product version and release number of its own. You are recommended to use CICSTSLEVEL to determine the version and release number of CICS Transaction Server.

RLSSTATUS(*cvda***)**
returns a CVDA value indicating whether VSAM RLS is active—that is, the CICS region is registered (with a currently-open control ACB) with an SMSVSAM address space. CVDA values are:

NOTAPPLIC
This CICS region does not support VSAM RLS because:

- CICS initialized with RLS=NO as a system initialization parameter, or
- CICS has forced RLS=NO because the level of VSAM in the MVS in which CICS is running does not support VSAM RLS.

RLSACTIVE
CICS has registered with an SMSVSAM server and VSAM RLS is currently active.

RLSINACTIVE
CICS has registered with an SMSVSAM server, but VSAM RLS is currently inactive due to an SMSVSAM server failure. All RLS requests fail until CICS performs dynamic VSAM RLS restart, which occurs automatically when the SMSVSAM server has restarted.

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 210).

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).

SDTRAN(*data-area***)**
returns the 4-character name of the transaction to be run at the beginning of normal or immediate shutdown. This may be the name of a user-supplied transaction, or the CICS-supplied default transaction, CESD.

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 169). CVDA values are:

CANCELLED
CICS is canceled.

CONTROLSHUT
CICS is performing a controlled shutdown (that is, a normal shutdown with a warm keypoint).

NOTAPPLIC
CICS is not shutting down.

SHUTDOWN
CICS is performing an immediate shutdown.

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 an initial or a cold start.

Note: The STARTUP option does not distinguish between an initial and a cold start. See the COLDSTATUS option.

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), *yy* is the low-order two digits of the year and *ddd* is the 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.

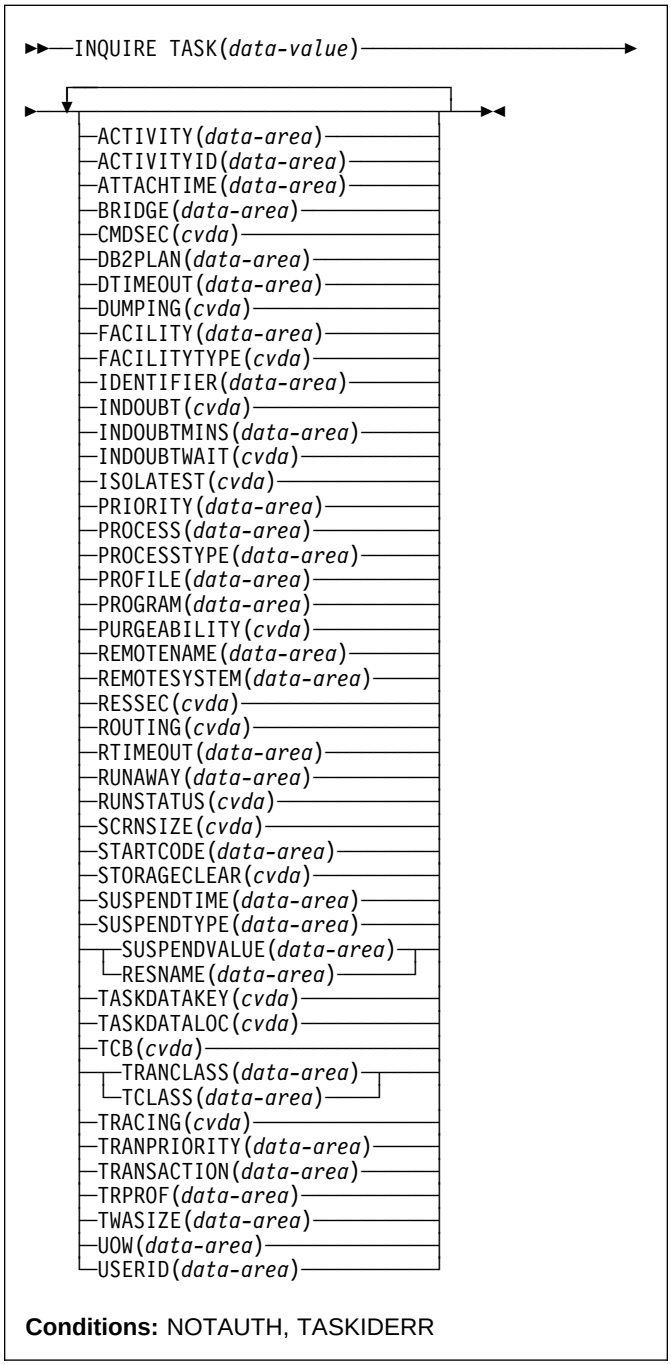
Conditions

NOTAUTH
RESP2 values:

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

INQUIRE TASK

Retrieve information about a user task.



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

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

ACTIVITY(data-area)
returns the 16-character, user-assigned, name of the CICS business transaction services activity that this task is executing on behalf of.

ACTIVITYID(data-area)
returns the 52-character, CICS-assigned, identifier of the CICS business transaction services activity that this task is executing on behalf of.

ATTACHTIME(*data-area***)**
returns an 8-byte packed decimal value, in ABSTIME format, representing the time in milliseconds at which the task was attached.

BRIDGE(*data-area***)**
returns the 4-character transaction identifier of the bridge monitor transaction that issued a START BREXIT TRANSID command to start this task. If the task is not currently running in the 3270 bridge environment, blanks are returned.

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 173 and the *CICS Resource Definition Guide* for more information).

DB2PLAN(*data-area***)**
returns a 1- to 8-character name of the DB2PLAN being used by this task, or blanks if no DB2PLAN is being used.

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.

IDENTIFIER(*data-area***)**
returns a 48-character field containing user data provided by the bridge exit, if the task was initiated in the 3270 bridge environment, or blanks, otherwise. This field is intended to assist in online problem resolution. For example, it could contain the MQ™ correlator for the MQ bridge, or a Web token.

INDOUBT(*cvda***)**
returns a CVDA value, based on the ACTION attribute of the TRANSACTION resource definition, indicating the action to be taken if the CICS region fails, or loses connectivity with its coordinator while a unit of work is in the in-doubt period.

The action is dependent on the values returned in INDOUBTWAIT and INDOUBTMINS; if INDOUBTWAIT returns WAIT, the action is not taken until the time returned in INDOUBTWAIT expires.

CVDA values are:

BACKOUT
: All changes made to recoverable resources are to be backed out.

COMMIT
: All changes made to recoverable resources are to be committed, and the unit of work marked as completed.

Note: If a program uses the obsolete DTB option, which was replaced by INDOUBT, a CVDA value of NOTSUPPORTED is returned.

INDOUBTMINS(*data-area***)**
returns a fullword binary field giving the length of time, in minutes, after a failure during the in-doubt period, before the task is to take the action returned in the INDOUBT field. The returned value is valid only if the unit of work is in-doubt and INDOUBTWAIT returns WAIT.

See also INDOUBT and INDOUBTWAIT.

INDOUBTWAIT(*cvda***)**
returns a CVDA value, based on the WAIT attribute of the TRANSACTION definition, indicating how a unit of work (UOW) is to respond if a failure occurs while it is in an in-doubt state. CVDA values are:

NOWAIT
: The UOW is not to wait, pending recovery from the failure. CICS is to take immediately whatever action is specified on the ACTION attribute of the TRANSACTION definition.

WAIT
: The UOW is to wait, pending recovery from the failure, to determine whether recoverable resources are to be backed out or committed.

For further information about the meaning of the ACTION and WAIT attributes of the TRANSACTION definition, see the *CICS Resource Definition Guide*.

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 173). 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).

PROCESS(data-area)
returns the 36-character name of the CICS business transaction services process of which this task is a part.

PROCESSTYPE(data-area)
returns the 8-character identifier of the type definition of the CICS business transaction services process of which this task is a part.

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.

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 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.

RESNAME(*data-area***)**
RESNAME, an alternative to SUSPENDVALUE, returns a 16-character resource name of tasks suspended on TS queues.

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 173 and the *CICS 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 RTIMOUT 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 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 Problem Determination Guide*.

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***)**
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 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.

TCB(*cvda***)**
returns a CVDA value indicating the type of TCB under which the task is running. The CVDA values are:

CKOPEN
The task is running under a CICS key open TCB (for example, a JVM (mode J8) TCB).

INTERNAL
The task is running under one of the CICS internal TCBs. An internal TCB can be one of the following:

- The concurrent mode (CO) TCB
- The file-owning mode (FO) TCB
- The resource-owning mode (RO) TCB
- The ONC/RPC mode (RP) TCB
- The sockets listener mode (SL) TCB
- The secure sockets layer mode (SO) TCB
- A sockets mode (S8) TCB
- The FEPI mode (SZ) TCB.

QR The task is running under the CICS QR TCB.

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 1 to 10. If the task does not belong to such a class, the value returned is zero. (See the TRANCLASS option for more information.)

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 Problem Determination Guide* and the description of the CETR transaction in the *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 DFHTCL*nn*, where *nn* is the 2-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.

UOW(*data-area***)**
returns the 16-character local identifier of the unit of work associated with this task.

USERID(*data-area***)**
returns the 8-character identifier of the user associated with the task.

Conditions

INVREQ
RESP2 values:

1 SUSPENDVALUE is specified, but significant characters are lost.
3 TCLASS is specified, but the task belongs to a named CLASS, not a numbered CLASS. The user should specify the TRANCLASS option.
10 The requested data is held on a data profile, but the data profile is not available.

NOTAUTH
RESP2 values:

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

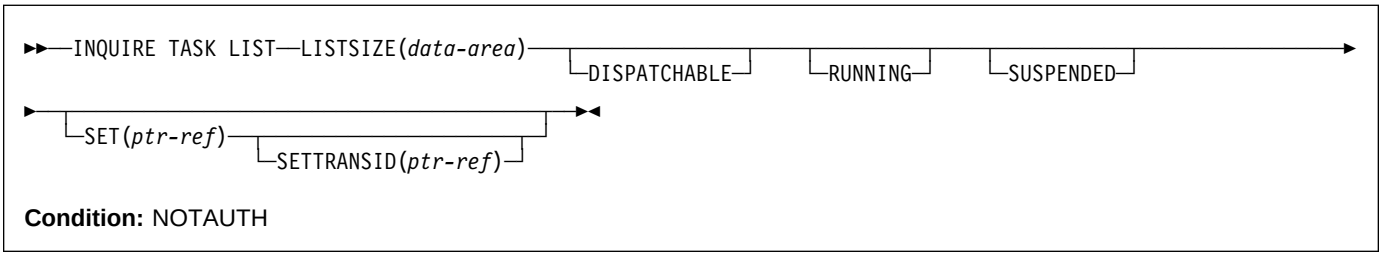
TASKIDERR
RESP2 values:

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.



Condition: NOTAUTH

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.

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).

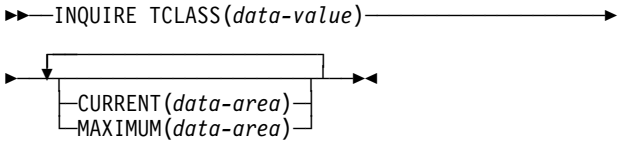
Conditions

NOTAUTH
RESP2 values:

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

INQUIRE TCLASS

Retrieve information about a transaction class.



Conditions: NOTAUTH, TCIDERR

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 “INQUIRE TRANCLASS” on page 205, 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:

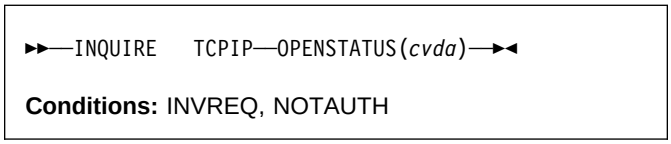
100 The user associated with the issuing task is not authorized to use this command.

TCIDERR
RESP2 values:

1 The named task class cannot be found.

INQUIRE TCPIP

Retrieve information about CICS internal sockets support.



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

Description

INQUIRE TCPIP returns information about the state of CICS internal sockets support.

Options

OPENSTATUS(*cvda***)**
returns a CVDA value indicating the status of CICS internal sockets support. CVDA values are:

OPEN CICS internal TCPIP support is open.

CLOSED CICS internal sockets support has not yet been activated, or has been terminated.

CLOSING CICS internal sockets support is in the process of closing.

IMMCLOSING CICS internal sockets support is in the process of immediate termination.

Conditions

INVREQ
RESP2 values:

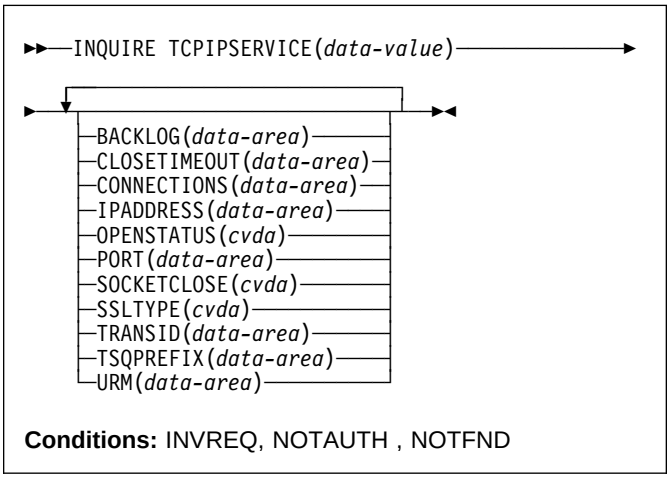
4 TCPIP=NO has been specified in the system initialization table.

NOTAUTH
RESP2 values:

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

INQUIRE TCPIPSERVICE

Retrieve information about the state of a service using CICS internal TCPIP support.



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

Description

INQUIRE TCPIPSERVICE allows you to retrieve information about TCPIP ports on which CICS internal TCPIP support is currently listening on behalf of other CICS services.

Options

BACKLOG(*data-area***)**
returns, in fullword binary form, the maximum number of requests which can be queued in TCP/IP waiting to be processed by the service.

CLOSETIMEOUT(*data-area***)**
returns, in fullword binary form, the number of seconds that have been specified for this service to wait for data for a new request. This can be between 0 and 86400 (24 hours).

CONNECTIONS
returns, in fullword binary form, the number of sockets connections for this service.

IPADDRESS(*data-area***)**
returns the 15-character dotted decimal IP address of this service.

OPENSTATUS(*cvda***)**
returns a CVDA value indicating the status of the CICS Web Interface. CVDA values are:

OPEN CICS internal sockets support is open for this service.

CLOSED CICS internal sockets support has not yet been activated, or has been terminated, for this service.

CLOSING CICS internal sockets support is in the process of closing for this service.

IMMCLOSING CICS internal sockets support is in the process of immediate termination.

PORT
returns, in fullword binary form, the number of the port on which CICS is listening on behalf of this service.

SOCKETCLOSE(cvda)
returns whether a TIMEOUT value is in effect for this service. CVDA values are:

WAIT NO was specified on the definition. Socket receives will wait for data indefinitely.

TIMEOUT A value was specified for the SOCKETCLOSE parameter on the definition. CLOSETIMEOUT returns the specified value.

SSLTYPE(cvda)
returns a CVDA value specifying the level of secure sockets support being used for this service. CVDA values are:

NOSSL Secure Sockets Layer is not being used for this service.

SSL Secure Sockets Layer without client authentication is being used for this service.

CLIENTAUTH Secure Sockets Layer with client authentication is being used for this service.

TCPIPSERVICE(*data-value***)**
specifies the 1- to 8-character name of the TCP/IP service about which you are inquiring.

TRANSID(*data-area***)**
returns the 4-character transaction id used on the attach for the task started to process a new request.

TSQPREFIX(*data-area***)**
returns the 6-character prefix of the temporary storage queue used to store inbound data and Web documents created by applications. The TS queue prefix must be matched by a corresponding TSMODEL definition to meet your system and application requirements.

URM(*data-area***)**
returns the 8-character name of the service user-replaceable module invoked by attached 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:

4 TCPIP not available (TCPIP=NO was specified as a system initialisation parameter)
5 TCPIP is closed.

NOTAUTH
RESP2 values:

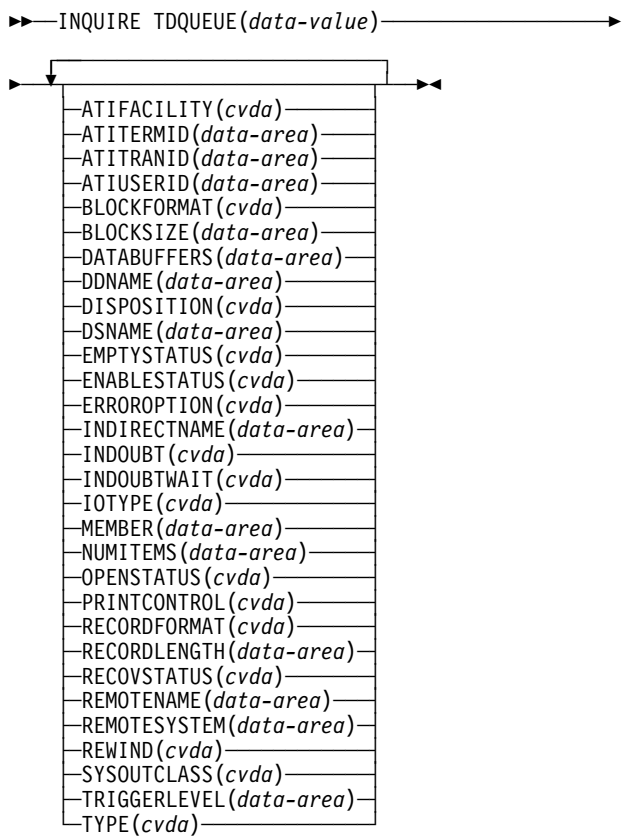
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

3 The TCPIPSERVICE was not found

INQUIRE TDQUEUE

Retrieve information about a transient data queue.



Conditions: ENDCOND, ILLOGIC, NORMAL, NOTAUTH, QIDERR

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.

You define transient data queues to CICS using transient data resource definitions. 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 “Null values” on page 13, are returned in such cases.

Two other types of queue exist: **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 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 18 for general information about browsing, including 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 that it creates to process the queue if no terminal associated with the queue. If the queue is not intrapartition, or no transaction is defined for it (the ATITRANID option), blanks are returned.

If the security manager is not active, the value returned is that of the default user ID and not any value that has been included in the installed definition.

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 an extrapartition queue.

UNBLOCKED
The records are not blocked.

BLOCKSIZE(*data-area*)
returns the length of the block in bytes (in the range 1 through 32767).

DATABUFFERS(*data-area*) (extrapartition queues only)
returns the number of buffers (in the range 1 through 255) to be used by the transient data queue.

DDNAME(*data-area*) (extrapartition queues only)
returns an 8-character identifier (padded with blanks if necessary) that may refer to a data set name used in the startup JCL.

DISPOSITION(*cvda*) (extrapartition queues only)
returns a CVDA value indicating the status of the associated data set. CVDA values are:

MOD
The system first assumes that the data set exists. For an existing data set, MOD causes the read/write mechanism to be positioned after the last record in the data set. The read/write mechanism is positioned after the last record each time the data set is opened for output.

If the system cannot find volume information for the data set on the DD statement, in the catalog, or passed with the data set from a previous step,

the system assumes that the data set is being created in this job step. For a new data set, MOD causes the read/write mechanism to be positioned at the beginning of the data set.

NOTAPPLIC
The option does not apply because the queue is not open or is not an extrapartition queue.

OLD The data set existed before this job step.

SHR The data set existed before this job step and can be read by other concurrent jobs.

DSNAME(*data-area***) (extrapartition queues only)**
returns a 1- to 44-character name that indicates an associated QSAM data set, or DUMMY data set. This is blank if SYSOUTCLASS is used.

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:

DISABLE PENDING
The queue is currently being disabled.

DISABLED
The queue cannot be accessed by applications. (For extrapartition queues, this value does not necessarily mean that the associated data set is closed.)

DISABLING
The queue is currently being disabled.

ENABLED
The queue can be accessed by applications.

NOTAPPLIC
The queue is indirect.

ERROROPTION(*cvda***) (extrapartition queues only)**
returns a CVDA value indicating the action that CICS should take if an I/O error is encountered. CVDA values are:

IGNORERR
The block that caused the error is accepted.

SKIP
The block that caused the error is skipped.

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.

INDOUBT(*cvda***) (intrapartition queues only)**
returns a CVDA value indicating the action CICS is to take for an in-doubt unit of work (UOW) if the definition for this queue specifies WAIT(YES). CVDA values are:

QUEUE
The UOW is in-doubt and waiting; any locks held by the UOW for this queue remain active until the final state of the UOW is known. This means that tasks are suspended rather than receiving the LOCKED response. When the final state of the UOW is known, any changes that it has made are committed or backed out. Until then, any further requests of the following types that need one of the active locks must wait:

- READQ if the in-doubt UOW had issued READQ or DELETEQ requests.
- WRITEQ if the in-doubt UOW had issued WRITEQ or DELETEQ requests.
- DELETEQ if the in-doubt UOW had issued READQ, WRITEQ, or DELETEQ requests.

REJECT
The UOW is in-doubt and waiting, and any locks held by the UOW for this queue are retained until the final state of the UOW is known. When the final state is known, any changes it has made are committed or backed out. Until then, any further requests that need one of the retained locks are rejected, and a LOCKED condition is returned. REJECT causes LOCKED to be raised in exactly the same circumstances as those in which QUEUE causes a transaction to wait.

INDOUBTWAIT(*cvda***) (intrapartition queues only)**
returns a CVDA value indicating whether an in-doubt unit of work (UOW), which has modified a recoverable queue, should wait for resynchronization with its coordinator to determine whether to commit or backout the changes. CVDA values are:

NOWAIT
The UOW is not to wait, and any changes made to recoverable resources are to be backed out or committed, as specified by the ACTION attribute on the transaction resource definition.

WAIT
The UOW is to wait and any action required while waiting is determined by the WAITACTION option.

This parameter overrides the WAIT option defined on the UOW's transaction definition. See the *CICS Resource Definition Guide* for an explanation of the interactions of in-doubt attributes on the TDQUEUE and TRANSACTION definitions.

IOTYPE(*cvda***) (extrapartition queues only)**
returns a CVDA value indicating whether the queue was defined for INPUT, OUTPUT, or RDBACK. 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.

RDBACK
The queue is defined for input and will be read backward.

MEMBER(*data-area***) (extrapartition queues only)**
returns the 8–character member name if the queue is a member of a partitioned data set. If not, blanks are returned.

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, or 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.

REWIND(*cvda*) (extrapartition queues only)
returns a CVDA value indicating the disposition of a tape data set. CVDA values are:

LEAVE
The current tape is positioned to the logical end of the data set.

REREAD
The current tape is positioned to reprocess the data set.

SYSOUTCLASS(*data-area*)
returns a single character indicating the class attribute of the associated SYSOUT data set (or blank if DSNAME is used).

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

ENDCOND
RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC
RESP2 values:

1 Browse sequence error.

NORMAL
RESP2 values:

0 No errors

NOTAUTH
RESP2 values:

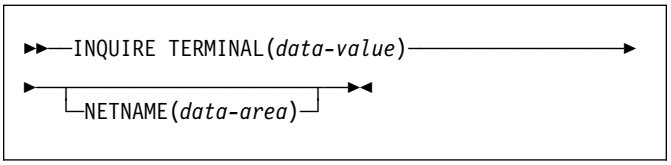
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

QIDERR
RESP2 values:

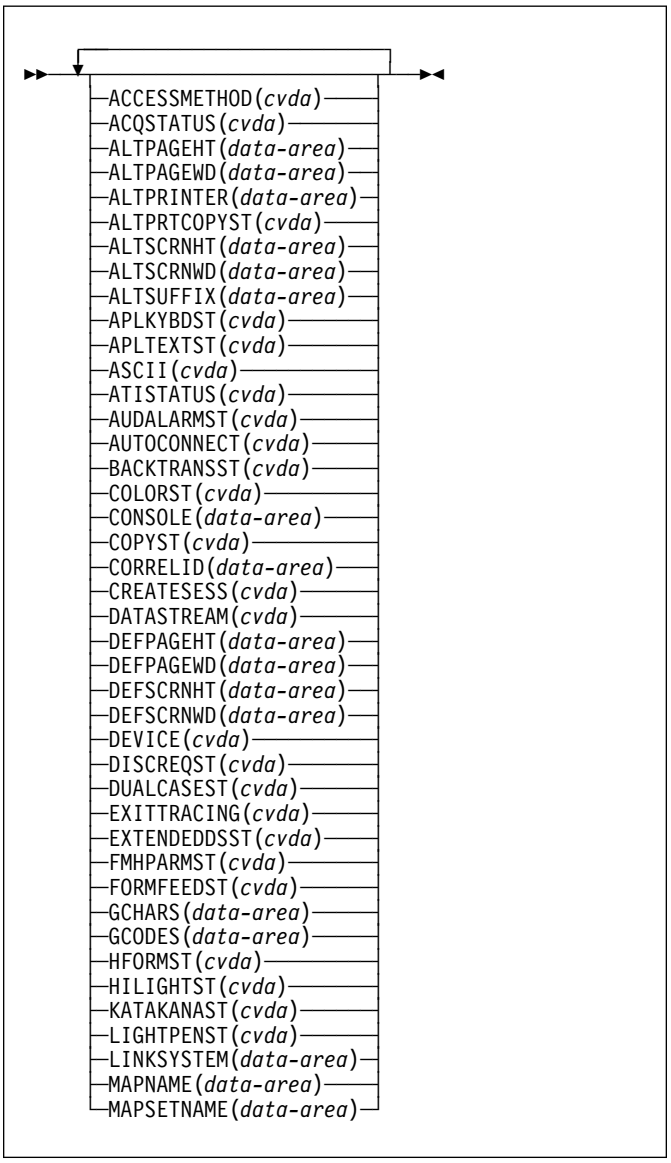
1 The named queue cannot be found.

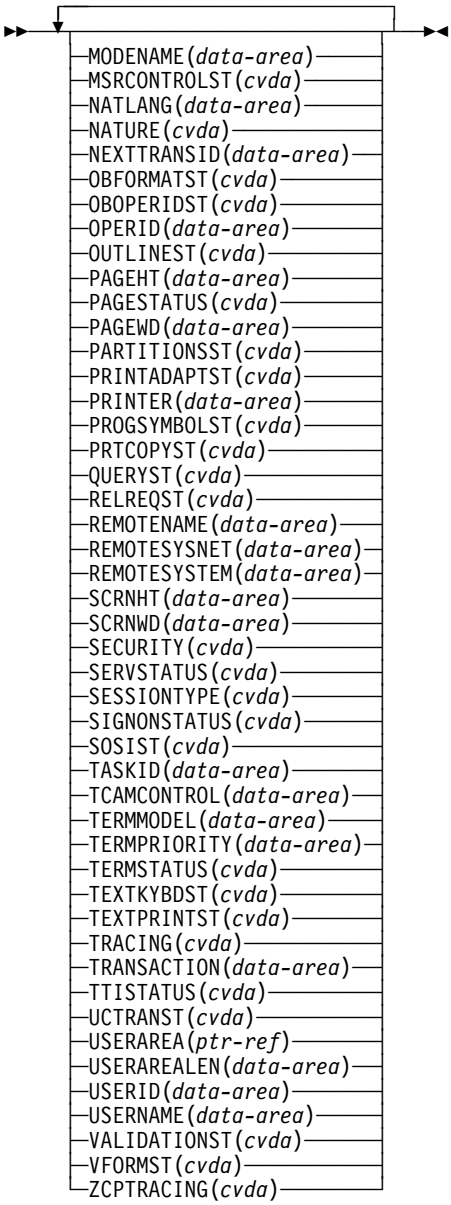
INQUIRE TERMINAL

Retrieve information about a terminal or session.



The following options apply to both the INQUIRE TERMINAL and the INQUIRE NETNAME command.





Conditions: END, ILLOGIC, NOTAUTH, TERMIDERR

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

Description

The INQUIRE TERMINAL and INQUIRE NETNAME commands both return information about a particular terminal or session installed in a CICS region.

You can use these commands to inquire about any type of terminal resource, including:

- Physical terminals owned locally (by the region in which the INQUIRE is 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)
- Models (definitions used only to autoinstall other terminals).
- MVS consoles defined to CICS.

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 details 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 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 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.

Remote terminals: In addition to links to local terminals or devices, some terminal definitions refer to remote terminals. A remote terminal is a terminal that is owned by another CICS system. The owning system is called the terminal-owning region (TOR). Note that different terminals may have different TORs. TOR here refers to the owning system for a particular terminal.

The way that the local system is connected to the TOR makes a difference to the information that the REMOTESYSTEM field returns.

If the TOR is **directly** connected to the local system, the REMOTESYSTEM field usually names the CONNECTION definition for the link. (It can name an indirect connection, but that is an unusual setup). In this case, the netname of the TOR is specified in the link CONNECTION definition.

If a remote terminal is on a system that is **not directly linked** to the TOR, the REMOTESYSTEM field can name one of two types of connection, as follows:

- A “real” connection that is the next link in the chain towards the TOR. In this case, the REMOTESYSNET option of the TERMINAL definition must specify the netname of the TOR.
- An indirect connection. In this case, the indirect connection NETNAME contains the netname of the TOR and its INDSYS option names another connection, which can also be indirect or “real.”

In both these cases, the LINKSYSTEM field of INQUIRE TERMINAL returns the “real” connection that is the next link towards the TOR. It is determined by looking at the logical chain of connections from the terminal in question to the “real” terminal entry. If the chain is broken (because a connection has not been installed yet, or has been discarded) LINKSYSTEM is not set. For a fuller explanation of the relationship between REMOTESYSTEM, REMOTESYSNET, and LINKSYSTEM, see “Remote connections” on page 107.

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 or INQUIRE NETNAME commands. See “Browsing resource definitions” on page 18 for general information about browsing, including 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.

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.

| 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.

CONSOLE(*data-area***)**
returns, for an MVS console only, a 12-byte string that identifies the console. If the device is not a console, CICS returns 12 blanks.

If the console is autoinstalled, or is defined explicitly with a console name, the name is returned in the first 8 bytes, and the last 4 bytes are blank.

If the console is defined by a numeric identifier, the string is divided into two sub-fields, separated by a period (.) in the ninth byte position. The sub-fields contain the following information:

- The first 8 bytes contain the MVS console name, if it is known, or the string '*UNKNOWN' if it isn't.
- The last 3 bytes contain the numeric console ID.

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.

CORRELID(*data-area***)**
returns an 8-character correlation-id that is only set for sessions, as follows:

- For LU6.1 sessions, it is set to the value of NETNAMEQ.
- For MRO sessions, it is set to the termid of the session at the other end of the MRO link to which this session is connected.
- For LU6.2 sessions, it is an 8-character token that is common to the two sessions that are connected.

Using CORRELID, you can relate the two parts of an MRO, LU6.1, or LU6.2 conversation, and so discover, for example, which program is running a particular function shipping mirror.

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 332.

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 203.) CVDA values are:

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.

EXTENDEDDSST(*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 EXTENDEDDS option) or implicitly, by specifying features that use the extended data stream (see the BACKTRANST, COLORST, HILIGHTST, MSRCONTROLST, OUTLINEST, PARTITIONSST, 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:

EXTENDEDDS
The terminal supports the extended data stream.

NOEXTENDEDDS
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.

HILIGHTST(*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:

HILIGHT
The terminal has extended highlighting.

NOHILIGHT
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.

LINKSYSTEM(*data-area***)**
returns the 4-character name of the connection that is the real link towards the TOR for a remote terminal entry, if it is available. It is not available if some connection definitions in the chain from the remote entry to the link system are missing.

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 Transaction Server for OS/390 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 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 Application Programming Reference* manual. Possible values are listed in the *CICS RACF Security Guide*. A blank means that no value has been specified.

NATURE(*cvda***)**
returns a CVDA value identifying the nature of the terminal definition. CVDA values are:

MODEL
A remote terminal definition (representing a terminal owned by another CICS region) which is not currently expanded into a surrogate.

REMSESSION
A remote session.

SESSION
A session.

SURROGATE
A remote terminal definition (representing a terminal owned by another CICS region) which is expanded into a surrogate.

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. For a remote terminal, it is the name by which the terminal is known to the VTAM in the remote region. (For a remote terminal routed from a pre-CICS Transaction Server for OS/390 region, NETNAME is blank.)

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 “INQUIRE NETNAME” on page 149.

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 is 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.

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:

NOOBOPERID
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 RACF Security Guide*. 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:

NOPRTCOPY
CICS is not to use the hardware copy feature.

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 an initial or 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.

REMOTESYSNET(*data-area***)**
returns the 8-character netname of the owning TOR, if the subject of this inquiry is a remote terminal. If it is blank, but the terminal is remote, the system named in the REMOTESYSTEM field has not been installed, and no value was specified for the REMOTESYSNET option when the terminal was defined.

REMOTESYSTEM(*data-area***)**
returns the 4-character name of a connection, if the subject of the inquiry is a remote terminal. The named connection can be either a connection entry that links towards the TOR, or an indirect connection that provides the netname of the TOR.

Otherwise this field is blank.

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.

TCAMCONTROL(*data-area*) (TCAM only)
returns a 1-character TCAM control byte giving one of the following codes to identify 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 the 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 uppercase 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 a 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 uppercase 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. See Table 5 on page 303 for information on how the UCTRAN options on the terminal and transaction profiles interact.)

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 uppercase. There is no difference in the data presented to the task between these two options.

UCTRAN
The input is translated to uppercase on receipt. (It is 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 user ID—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:

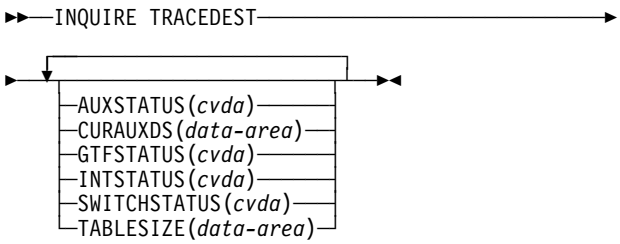
100 The user associated with the issuing task is not authorized to use this command.

TERMIDERR
RESP2 values:

1 The named terminal cannot be found.

INQUIRE TRACEDEST

Retrieve information about tracing.



Conditions: NOTAUTH

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 is 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.

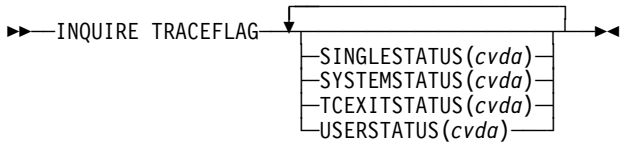
Conditions

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.



Conditions: NOTAUTH

For more information about the use of CVDAs, see "CICS-value data areas (CVDAs)" 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 Problem Determination Guide*.

Options

SINGLESTATUS(*cvda***)**
returns a CVDA value indicating whether tracing is turned on or is 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

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.

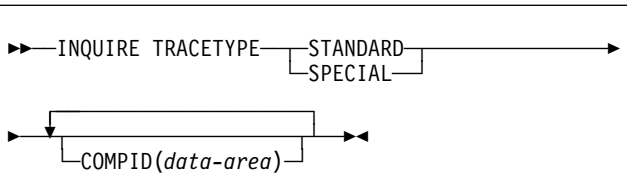
Conditions

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.



Conditions: NOTAUTH, NOTFND

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 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
BR	BRIDGE	3270 Bridge
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
KC		Task control
KE	KERNEL	Kernel
LC	LOCALCATLG	CICS local catalog manager
LD	LOADER	Program load manager
LG	LOGMGR	Log manager
LM	LOCKMGR	Lock manager
ME	MESSAGE	Message manager
MN	MONITOR	Monitoring manager
NQ		Enqueue domain
PA	PARAMGR	Parameter manager
PC		Program control
PG	PROGMGR	Program manager
RI		Resource manager interface (RMI)
RM		Recovery 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
WB	WEB	Web 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:

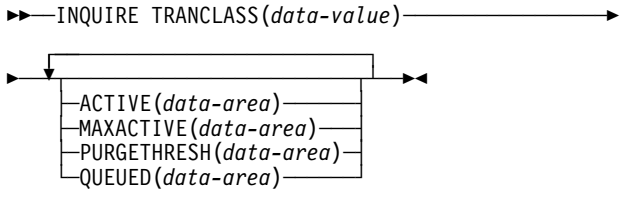
100 The user associated with the issuing task is not authorized to use this command.

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.



Conditions: INVREQ, NOTAUTH, TCIDERR

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 18 for general information about browsing, including 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).

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 DFHTCL*nn*, where *nn* is the two-digit class number.

Conditions

INVREQ
RESP2 values:

12 The TRANCLASS definition is in use.

NOTAUTH
RESP2 values:

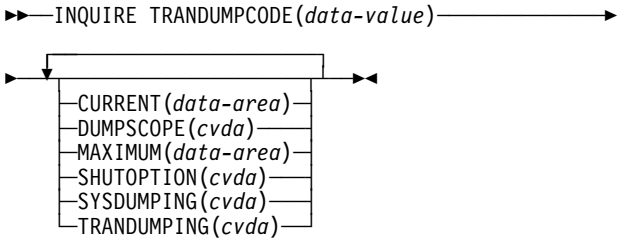
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

TCIDERR
RESP2 values:

1 The transaction class cannot be found.

INQUIRE TRANDUMPCODE

Retrieve information about a transaction dump code.



Conditions: END, ILLOGIC, NOTAUTH, NOTFND

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), and the number of 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 TRANSDUMPCODE command or the CEMT transaction; such changes are preserved over executions of CICS, until an initial or 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 an initial or 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 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 18 for general information about browsing, including 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 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 is 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:

NOSYSDUMP
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 is 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.

NOTAUTH
RESP2 values:

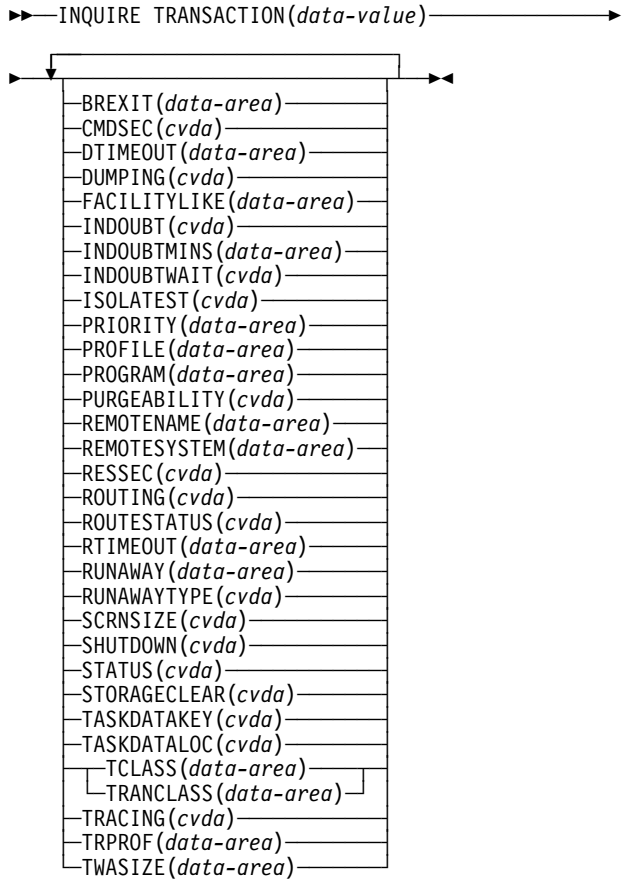
100 The user associated with the issuing task is not authorized to use this command.

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.

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 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 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

BREXIT(*data-area***)**
returns the 8-character name of the bridge exit defined by the BREXIT parameter of the named transaction resource definition.

If BREXIT is not defined, blanks are returned.

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.

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.

FACILITYLIKE(*data-area***)**
returns the 4-character name of the terminal defined by the FACILITYLIKE parameter in the PROFILE associated with the named transaction resource definition.

If FACILITYLIKEis not defined, blanks are returned.

INDOUBT(*cvda***)**
returns a CVDA value, based on the ACTION attribute of the TRANSACTION resource definition, indicating the action to be taken if the CICS region fails or loses connectivity with its coordinator while a unit of work is in the in-doubt period.

The action depends on the values returned in INDOUBTWAIT and INDOUBTMINS; if INDOUBTWAIT returns WAIT, the action is not normally taken until the time returned in INDOUBTMINS expires. (For exceptions to this rule, see INDOUBTWAIT.)

CVDA values are:

BACKOUT
All changes made to recoverable resources are to be backed out.

COMMIT
All changes made to recoverable resources are to be committed, and the unit of work marked as completed.

Note: If a program uses the obsolete DTB option, which was replaced by INDOUBT, a CVDA value of NOTSUPPORTED is returned.

INDOUBTMINS(*data-area***)**
returns a fullword binary field giving the length of time, in minutes, after a failure during the in-doubt period, before the transaction is to take the action returned in the INDOUBT field. The returned value is valid only if the unit of work is in-doubt and INDOUBTWAIT returns WAIT.

INDOUBTWAIT(*cvda***)**
returns a CVDA value, based on the WAIT attribute of the TRANSACTION definition, indicating how CICS is to respond if a failure occurs while a unit of work (UOW) is in an in-doubt state. CVDA values are:

NOWAIT
The UOW is not to wait, pending recovery from the failure. CICS is to take immediately whatever action is specified on the ACTION attribute of the TRANSACTION definition.

WAIT
The UOW is to wait, pending recovery from the failure, to determine whether recoverable resources are to be backed out or committed.

Note: Even if INDOUBTWAIT returns WAIT, there may be aspects of the UOW that force CICS to take an immediate decision—that is, to take immediately the action specified on the ACTION attribute of the transaction definition. This can happen if, for example, the UOW contains:

- Subordinate LU6.1 sessions
- Subordinate MRO sessions to pre-CICS Transaction Server for OS/390 systems.

For further information about the meaning of the ACTION and WAIT attributes of the TRANSACTION definition, see the *CICS Resource Definition Guide*.

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:

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 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.

ROUTESTATUS
returns a CVDA value indicating whether, if the transaction is the subject of an eligible START command, it will be routed using the enhanced routing method. CVDA values are:

NOTROUTABLE
If the transaction is the subject of a START command, it will be routed using the “traditional” method.

ROUTABLE
If the transaction is the subject of an eligible START command, it will be routed using the enhanced method.

For details of the enhanced and "traditional" methods of routing transactions invoked by EXEC CICS START commands, see the *CICS Intercommunication Guide*.

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 1 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 DFHTCL*nn*, 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.

TRANCLASS(*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.

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:

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.

NORMAL
RESP2 values:

10 The profile definition associated with the transaction is not available.

NOTAUTH
RESP2 values:

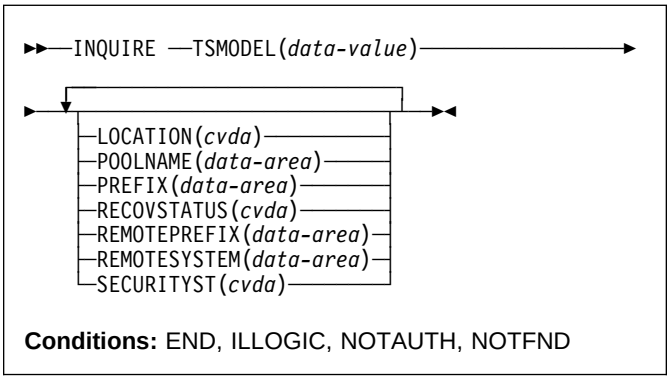
100 The user associated with the issuing task is not authorized to use this command.
101 The user associated with the issuing task is not authorized to access this particular resource in the way required by this command.

TRANSIDERR
RESP2 values:

1 The transaction could not be found.

INQUIRE TSMODEL

Retrieve information about a temporary storage model.



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

Description

The INQUIRE TSMODEL command returns information about a particular TS model.

Browsing

You can also browse through all of the temporary storage models in your system by using the browse options (START, NEXT, and END) on INQUIRE TSMODEL commands.

See "Browsing resource definitions" on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

LOCATION(*cvda***)**
returns a CVDA value indicating where queues matching the model are to be stored. CVDA values are:

AUXILIARY
Queues matching this model are to be held on auxilairy storage.

MAIN
Queues matching this model are to be held in main storage.

POOLNAME(*data-area***)**
returns an 8-character shared pool name.

PREFIX(*data-area***)**
returns a 16–byte character string, or a 32–byte hex string with the value of the prefix for this model.

RECOVSTATUS(*cvda***)**
returns a CVDA value indicating the recovery status for this model. CVDA values are:

RECOVERABLE
Queue names matching this model are recoverable.

NOTRECOVABLE
Queue names matching this model are non-recoverable.

REMOTEPREFIX(*data-area***)**
returns the 16–byte character string, or 32–byte hex string to be used as the name prefix on the remote system.

REMOTESYSTEM(*data-area***)**
returns the 4-character name of the remote system on which the queues matching this model will be defined.

SECURITYST(*cvda***)**
returns a CVDA value indicating the security status for this model. CVDA values are:

SECURITY
Security checking will be performed for queue names matching this model.

NOSECURITY
Security checking will not be performed for queue names matching this model.

TSMODEL(*data-value***)**
specifies the 8-character name of a temporary storage 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.

NOTAUTH
RESP2 values:

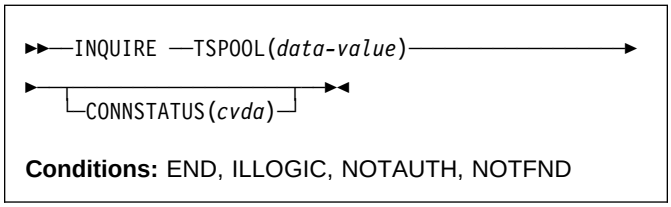
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The TSMODEL does not exist.

INQUIRE TSPOOL

Retrieve information about a shared temporary storage pool.



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

Description

The INQUIRE TSPOOL command returns information about a particular shared temporary storage pool.

Browsing

You can also browse through all of the temporary storage pools in your system by using the browse options (START, NEXT, and END) on INQUIRE TSPOOL commands.

See "Browsing resource definitions" on page 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

TSPOOL(*data-value***)**
returns an 8-character field giving the shared TS pool name.

CONNSTATUS(*cvda***)**
returns a CVDA value containing the connection status of this pool. CVDA values are:

CONNECTED
This pool is connected.

UNCONNECTED
This pool is not connected.

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.

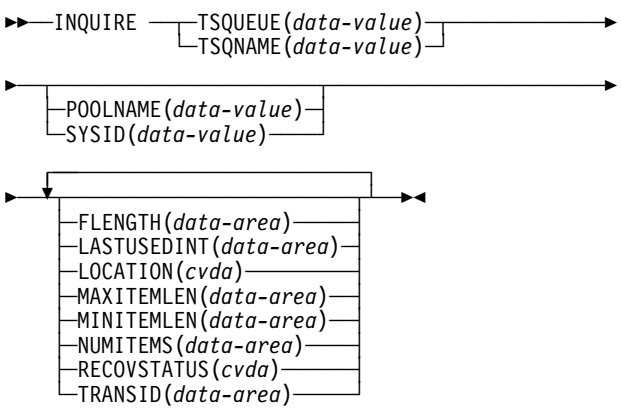
NOTFND
RESP2 values:

1 The TSPOOL does not exist.

INQUIRE TSQUEUE / TSQNAME

Retrieve information about a temporary storage queue.

| This section applies also to the alternative command,
| INQUIRE TSQNAME Use either to Inquire about names up
| to 8 characters long, use INQUIRE TSQNAME to Inquire
| about names up to 16 characters long.



Conditions: END, ILLOGIC, INVREQ, NOTAUTH, POOLERR, QIDERR, SYSIDERR

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 eight characters with nulls (X'00'). If you want to browse TS queues that are in a shared temporary storage pool managed by a TS server, you must specify the POOLNAME or the SYSID option on the browse START request only. If CICS cannot find the specified SYSID in any temporary storage table (TST) TYPE=SHARED entry, CICS returns the INVREQ condition.

Note: If you do a WRITEQ, for example, to queue, which maps to a shared TS pool because of a TST definition, be aware that to inquire on this queue you need to specify the explicit SYSID on the INQUIRE command.

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 18 for general information about browsing, including syntax, exception conditions, and examples.

Options

FLENGTH(*data-area***)**
returns a fullword binary field giving the total length in bytes of all the items in the temporary storage queue. For more information about queue lengths, see the MAXITEMLEN option.

For shared queues only: When the whole shared queue has been stored in a single entry in the coupling facility (in which case the returned value for FLENGTH is less than 32K (32768)), this is the total size of all items including their control information.

When the shared queue has been stored as a separate list in the coupling facility, the total size is estimated as MAXITEMLEN times NUMITEMS.

LASTUSEDINT(*data-area***)**
returns a fullword binary field specifying the interval in seconds since the temporary storage queue was last referenced.

The value returned for 'large' shared temporary storage queues is governed by the value of the LASTUSEDINTERVAL parameter specified for the associated TS queue manager (see the *CICS System Definition Guide* for further information).

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 (or in the coupling facility).

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 8 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.

For auxiliary temporary storage, the length is rounded to the next highest multiple of either 64 or 128 (depending on the control interval size of the temporary storage data set). (For background information about CI sizes, see the *CICS System Definition Guide*.)

For shared queues, the lengths returned in MINITEMLEN, MAXITENLEN, and FLENGTH, reflect the data length stored in the coupling facility. This includes any item control information, which consists of a 2-byte length prefix for each item.

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, and for information about shared TS queues, see the MAXITEMLEN option.

NUMITEMS(*data-area***)**
returns a halfword binary field giving the number of items in the temporary storage queue.

POOLNAME(*data-value***) (TS data sharing only)**
specifies the name of a temporary storage pool. CICS ships the command to the temporary storage server that manages the pool.

For browse operations, specify POOLNAME on the browse START request only, not on the NEXT or END requests.

RECOVSTATUS(*cvda***)**
returns a CVDA value indicating the recovery status of the queue. CVDA values are:

RECOVERABLE
The queue is recoverable.

NOTRECOVERABLE
The queue is not recoverable.

SYSID(*data-value***) (TS data sharing only)**
specifies the system name that corresponds to a temporary storage pool name. If CICS finds the specified system name in a TST TYPE=SHARED entry, it ships the command to the temporary storage server that manages the pool.

For browse operations, specify SYSID on the browse START request only, not on the NEXT or END requests.

TRANSID(*data-value***)**
specifies the identifier of the transaction which created the temporary storage queue.

TSQUEUE(*data-value***)**
specifies the 8-character name of the temporary storage queue about which you are inquiring.

TSQNAME(*data-value***)**
is an alternative to TSQUEUE and specifies the 16-character name of the temporary storage queue 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.

INVREQ
RESP2 values:

1 The specified SYSID does not exist in any TYPE=SHARED entry in the temporary storage table.
2 When INQUIRE TSQUEUE NEXT is specified, the NEXT queue to be browsed has a queue name of more than 8 significant characters. The queue name is truncated, some significant characters are lost.
4 This temporary storage queue name cannot be deleted as it was written by CICS using the PUTQ macro.

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.

POOLERR
RESP2 values:

3 The POOLNAME does not exist.

QIDERR
RESP2 values:

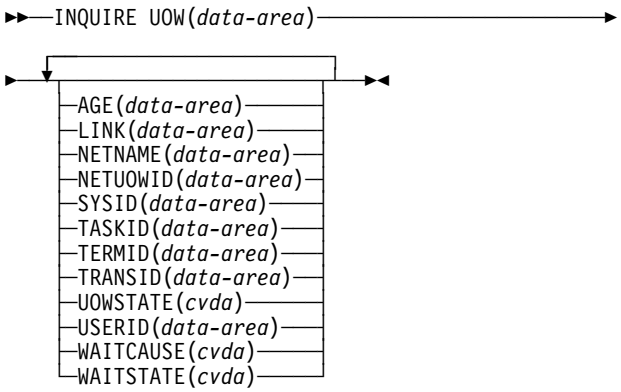
1 The temporary storage queue cannot be found.

SYSIDERR
RESP2 values:

3 The SYSID does not map to a shared pool.
4 Server error.
5 I/O error on coupling facility.

INQUIRE UOW

Retrieve information about a unit of work (UOW).



Conditions: END, ILLOGIC, NOTAUTH, UOWNOTFOUND

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

Description

The INQUIRE UOW command retrieves information about a unit of work, or about all UOWs in a specified state. It returns the state of the UOW (for example, INDOUBT) and whether it is active, waiting, or shunted. In some cases, it returns the name of the resource that caused the UOW to be shunted, plus the transaction, user, and terminal that started it.

Important: In an intercommunication environment, a unit of work can include actions that are to be taken by two or more connected systems. Such a unit of work is known as a *distributed* unit of work, because the resources to be updated are distributed across more than one system. A distributed unit of work is made up of two or more *local* units of work, each of which represents the work to be done on one of the participating systems.

Note that INQUIRE UOW always returns information about *local* UOWs—that is, for a distributed UOW it returns information only about the work required on the system on which the command is issued. You can assemble information about a distributed UOW by matching the network-wide UOW identifier returned in the NETUOWID field against the network-wide identifiers of local UOWs on other systems.

For further information about local and distributed UOWs, see the *CICS Intercommunication Guide*.

Browsing

You can also browse through all of the UOWs currently in your system by using the browse options (START, NEXT, and END) on INQUIRE UOW commands. See "Browsing resource definitions" on page 18 for general information about browsing, including syntax, exception conditions, and examples.

For example, if you suspect a problem with either a recoverable data set or a connection, you can use INQUIRE UOW to return information about UOWs that have been shunted because of a connection or data set failure.

Note: Do not issue SYNCPOINT commands during an INQUIRE UOW browse. The following sequence of commands causes an infinite loop:

```
EXEC CICS INQUIRE UOW START
EXEC CICS INQUIRE UOW(data-area) NEXT
SYNCPOINT
EXEC CICS INQUIRE UOW(data-area) NEXT
SYNCPOINT
⋮
EXEC CICS INQUIRE UOW END
```

This is because every time the SYNCPOINT command is executed, a new UOW is created. The new UOW is returned in the next INQUIRE UOW, which is followed by a SYNCPOINT, and so on.

For examples of the use of the INQUIRE UOW command, see the *CICS Problem Determination Guide*.

Options

AGE(*data-area***)**
returns a fullword binary value giving the number of seconds since the UOW entered its current WAITSTATE.

LINK(*data-area***)**
returns an 8-character value that, for a WAITCAUSE value of CONNECTION, is the netname of the remote system that caused the UOW to wait or be shunted. For other WAITCAUSE values, LINK returns blanks.

NETNAME(*data-area***)**
returns the 8-character network name of the terminal from which the UOW was started.

If the UOW was started from an ISC or MRO session, NETNAME returns the network name of the remote region.

If the UOW was not started from a terminal, nor from an ISC or MRO session, NETNAME returns blanks.

NETUOWID(*data-area***)**
returns the LU6.2 name for the UOW within this network—that is, the network-wide identifier of the UOW. This is a 27-character data-area.

You can assemble information about a distributed UOW by matching the network-wide UOW identifier against the network-wide identifiers of local UOWs on other systems.

SYSID(*data-area***)**
returns a 4-character value that, for a WAITCAUSE value of CONNECTION, is the sysid of the connection that caused the UOW to wait or be shunted. If the connection has been discarded, and for other WAITCAUSE values, SYSID returns blanks.

TASKID(*data-area***)**
returns a 4-byte packed-decimal value giving the task number originally associated with this UOW. If the UOW is shunted, the task terminates. In this case, the number may have been reused by another task.

TERMID(*data-area***)**
returns the 4-character ID of the terminal or session from which this UOW was started. This is the principal facility for the task. If the transaction is the mirror transaction, CSMI, it is the session.

TRANSID(*data-area***)**
returns the 4-character ID of the transaction that started this UOW.

UOW(*data-area***)**
specifies the 16-character local identifier of the UOW about which you are inquiring, the last eight bytes of which are always null (X'00').

UOWSTATE(*cvda***)**
returns a CVDA value indicating the state of the UOW. CVDA values are:

BACKOUT
This UOW is being backed out, or has failed to back out one or more of the recoverable resources involved in the UOW.

COMMIT
A decision to commit the UOW has been made, but the UOW is waiting or has been shunted. This may be because the decision has not yet been communicated to all participants in the syncpoint, or because a failure has occurred during commit processing.

| FORCE
| An attempt is being made to force the UOW to back out or commit, as specified on the ACTION option of the TRANSACTION resource definition.

HBACKOUT
The UOW has been forcibly backed out. A forced decision is taken when a UOW is unable to wait for in-doubt resolution—for example, the transaction may have been defined as WAIT(NO), or backed out with a CEMT SET UOW command.

HCOMMIT
The UOW has been forcibly committed.

INDOUBT
This UOW is in the in-doubt state.

INFLIGHT
The UOW is running normally.

USERID(*data-area***)**
returns the 8-character user ID for which this transaction was running.

WAITCAUSE(*cvda***)**
returns a CVDA value identifying the type of resource that caused the UOW to wait or be shunted.

Note: In the case of a wait, it is the UOW that is waiting, not the task.

Because each resource needs fields of the right type, WAITCAUSE also indicates which fields contain the RESOURCE NAME and QUALIFIER. CVDA values are:

CONNECTION
This UOW is waiting or has been shunted because of the failure of a session to the coordinator of the UOW during the in-doubt period. NETNAME and SYSID contain the netname and system name of the failed link.

DATASET
This UOW is waiting or has been shunted because of a failure associated with one or more data sets. Use the INQUIRE UOWDSNFAIL command to identify the data sets involved and the reasons why they have caused the UOW to fail.

RLSSERVER
This UOW is waiting or has been shunted because of the failure of an RLS server.

| RRMS
| This UOW is waiting or has been shunted because communication has been lost with RRS/MVS.

WAITCOMMIT
This UOW is waiting or has been shunted because a failure occurred during commit processing.

WAITFORGET
This UOW is waiting for FORGET from participants in the syncpoint. Use the INQUIRE UOWLINK command to obtain the netnames and sysids of the participants.

WAITRMI
This UOW is waiting for FORGET from the RMI. Use the INQUIRE UOWLINK command to obtain the entry name and qualifier of the task-related user exit.

WAITSTATE(*cvda***)**
returns a CVDA value indicating whether the UOW is currently running or waiting. CVDA values are:

ACTIVE
: The UOW is running normally.

SHUNTED
: Syncpoint processing of the UOW has been deferred. A reason for this is returned in WAITCAUSE. SHUNTED further indicates that the task, terminal and program storage have been released, and locks have been retained.

WAITING
: Syncpoint processing has completed on this system, but not on all systems involved in the distributed UOW. WAITCAUSE returns either WAITFORGET or WAITRMI, and UOWSTATE returns either BACKOUT or COMMIT to indicate how the UOW was resolved on this system.

Conditions

END
RESP2 values:

2 All authorized resource definitions have been retrieved. All data areas specified on this command are left unchanged.

ILLOGIC
RESP2 values:

1 A browse of this resource type is already in progress, or an INQUIRE UOW START command has not been issued.

NOTAUTH
RESP2 values:

100 The use of this command is not authorized.

UOWNOTFOUND
RESP2 values:

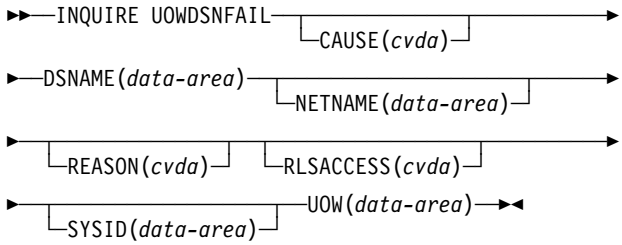
1 The named UOW cannot be found.

INQUIRE UOWDSNFAIL

Retrieve information about UOWs that have updated CICS file control-managed data sets.

Browse only function

The INQUIRE UOWDSNFAIL command can be used only in browse mode.



Conditions: END, ILLOGIC, NOTAUTH

Description

This command enables you to inquire on the reasons why UOWs were shunted because of a failure during syncpoint associated with a specified data set. Failures during syncpoint processing result in locks held by the UOW against the data set (or data sets) which suffered the failure being retained. Thus, when a failure is reported by this command, it also indicates the presence of retained locks.

The UOWDSNFAIL command returns UOWs that are shunted and also UOWs that are in the process of being retried. In the latter case, the only data sets returned are those that have not yet been processed as part of the retry.

Note that there may be failures against the data set by other CICS regions. The command needs to be issued on all regions in the sysplex to get a full picture of the state of the data set. See the *CICS Recovery and Restart Guide* for information about the CICS batch-enabling sample programs that assist you in doing this, and about the AMS SHCDS LIST subcommands that allow you to investigate retained locks held by CICS regions that are down.

Browsing

You can use the browse options (START, NEXT, and END) to find all the units of work with syncpoint failures, together with the data sets that have suffered failures. In addition, the reason is given for each unique UOW/data set combination (a UOW can have syncpoint failures for several data sets but, for each data set within the UOW, the cause of the failure is the same). See “Browsing resource definitions” on

page 18 for general information about browsing, including syntax, exception conditions, and examples.

Because this command returns information about UOWs that are currently failed with respect to data sets (with associated retained locks held against those data sets), it does not return information about failures that are in the process of being retried when the command is issued. For example, if a UOW suffered a backout failure with respect to a particular data set, and a SET DSNAME RETRY command was issued for that data set, that particular UOW/data set combination would not appear in the browse. The backout retry might either be successful, in which case the failure condition will have been cleared, or it might fail again, in which case the UOW/data set combination would appear if a new INQUIRE UOWDSNFAIL browse were started.

One important use of this command is to enable you to write a transaction that helps operators to identify and remove retained locks, so that data sets can be quiesced and used for batch application programs. There are several CICS-supplied sample programs that you can use unmodified, or use as a basis for writing your own programs. See the sample application programs, DFH0BAT1 through DFH0BAT8, for a working illustration of the use of this command. These are supplied in the CICSTS13.CICS.SDFHSAMP library, and are described in the *CICS Operations and Utilities Guide*.

The INQUIRE UOWDSNFAIL function is in effect a two dimensional, or nested, browse: the first (outer) browse loops through all the UOWs, and within each UOW, the second (inner) browse loops though all the failed datasets associated with that UOW. Note that, in common with all browse functions, CICS does not lock resources during a browse operation. For each failed UOW, CICS obtains a snapshot of all the data sets that are failed for the UOW, and returns one UOW/data set pair for each NEXT operation. It is theoretically possible that the status of some data sets associated with an INQUIRE UOWDSNFAIL NEXT command could have changed by the time the information is returned to your program.

Options

CAUSE(*cvda***)**
returns a CVDA value indicating which failed component has caused the UOW to have retained locks for this data set. CVDA values are:

CACHE
A VSAM RLS cache structure, or connection to it, has failed.

CONNECTION
An intersystem connection error has caused the UOW to fail while in-doubt. The name of the system to which connectivity was lost is returned on the SYSID parameter and its netname is returned on the NETNAME parameter. CICS returns additional information in the REASON parameter about the connection failure.

DATASET
The backout of a UOW has failed for this data set. The reason for the data set failure is returned in the REASON parameter.

RLSSERVER
The SMSVSAM server has failed. The reason for the data set failure is returned in the REASON parameter.

UNDEFINED
The UOW is probably being retried. This can occur following a SET DSN RETRY command, or automatically when the failed resource returns. It can also occur following an emergency restart.

DSNAME(*data-area***)**
returns, as a 44-character value, the data set name of a data set that has suffered a backout failure in this UOW.

NETNAME(*data-area***)**
returns the 8-character netname (when the CVDA on the CAUSE parameter is CONNECTION) of the remote system to which connectivity has been lost.

REASON(*cvda***)**
returns a CVDA value (when the CVDA returned on the CAUSE parameter is RLSSERVER, CONNECTION, or DATASET) indicating the specific reason for the error against this data set. CVDA values are:

BACKUPNONBWO
Backout of the updates made to the data set by the UOW failed because a non-BWO backup of the data set was in progress while the UOW was being backed out. When the backup completes, CICS automatically retries the UOW.

COMMITFAIL
An error occurred at some point when RLS locks were in the process of being released. This is an error that can normally be resolved by recycling the SMSVSAM server (which should happen automatically). The locks were acquired as a result of recoverable requests having been issued against the data set.

DATASETFULL
No space is available on the direct access device for adding records to a data set. You need to reallocate the data set with more space. You can then retry the backout using SET DSNAME RETRY. For further information, see the section on moving data sets in the *CICS Recovery and Restart Guide*.

DEADLOCK (non-RLS data sets only)
A deadlock was detected during backout. This is a transient condition that will probably go away if the backout is retried.

DELEXITERROR
: Backout of a write to an ESDS failed because a logical delete global user exit program was not enabled, or a logical delete global user exit program decided not to execute the logical delete.

FAILEDBKOUT
: This occurs as a result of a severe error being identified during backout, and is possibly an error in either CICS or VSAM. The problem may go away if the backout is retried. Note that CICS performs some first-failure data capture (FFDC) at the point where the error is first detected.

INDEXRECFULL
: A larger alternate index record size needs to be defined for the data set. For further information, see the section on moving data sets in the *CICS Recovery and Restart Guide*.

  This error can also occur when a unique alternate index key, for a non-RLS data set, has been reused and CICS is now backing out the request which had removed that key value.

INDOUBT
: The unit of work had issued recoverable requests against the data set, and has now failed in-doubt. The connection to the coordinating system needs to be reestablished.

IOERROR
: A hard I/O error occurred during backout. To correct this error, restore a full backup copy of the data set and perform forward recovery. If you use CICSVR as your forward recovery utility, the backout is automatically retried for an RLS data set. For a non-RLS data set, use the SET DSNAME (...) RETRY command to drive the backout retry.

LCKSTRUCFULL
: An attempt to acquire a lock during backout of an update to this data set failed because the RLS lock structure was full. You must allocate a larger lock structure in an available coupling facility and rebuild the existing lock structure into it, then use the SET DSNAME (...) RETRY command to drive the backout retry.

NOTAPPLIC
: The CVDA for CAUSE is not CONNECTION, RLSSERVER, or DATASET.

OPENERROR
: Error on opening the file for backout. A console message notifies you of the reason for the open error. One likely reason could be that the data set was quiesced.

RLSGONE
: An error occurred when backing out the UOW, because the SMSVSAM RLS server was inactive. This may also be the reason why the UOW went into backout originally. This is an error that can be resolved by recycling the server (which should happen automatically). Generally, when the server recovers, the UOWs are retried automatically. In very exceptional circumstances, it may be necessary to issue a SET DSNAME(...) RETRY command to retry UOWs that were not retried when the server returned.

RRCOMMITFAIL
: An error occurred while RLS locks for the unit of work were being released. For this data set, the locks being released were all repeatable read locks, so if the failure was due to the RLS server being unavailable, the locks will have been released. If the failure was due to some other error from the SMSVSAM server, the locks may still be held.

RRINDOUBT
: The unit of work had issued repeatable read requests against the data set, and has now failed with an in-doubt condition. The locks will have been released, so this failure does not prevent you from running a batch job against the data set. However, if you want to open the data set in non-RLS mode from CICS, you need to resolve the in-doubt failure before you can define the file as having RLSACCESS(NO). If the unit of work has updated any other data sets, or any other resources, you should try to resolve the in-doubt failure correctly. If the unit of work has only performed repeatable reads against VSAM data sets and has made no updates to other resources, it is safe to force the unit of work using the SET DSNAME or SET UOW commands.

Each REASON (except for NOTAPPLIC) corresponds with only one CAUSE value. The mappings are as follows:

Cause	Reason
CACHE	NOTAPPLIC
CONNECTION	INDOUBT
CONNECTION	RRINDOUBT
DATASET	BACKUPNONBWO
DATASET	DELEXITERROR
DATASET	DATASETFULL
DATASET	DEADLOCK
DATASET	FAILEDBKOUT
DATASET	INDEXRECFULL
DATASET	LCKSTRUCFULL
DATASET	IOERROR
DATASET	OPENERROR
RLSSERVER	COMMITFAIL
RLSSERVER	RRCOMMITFAIL
RLSSERVER	RLSGONE
UNDEFINED	NOTAPPLIC

RLSACCESS(*cvda***)**
: returns a CVDA value indicating whether the data set was last opened in this CICS region in RLS or non-RLS mode. CVDA values are:

NOTRLS
The last open in this CICS region was in non-RLS mode.

RLS The last open in this CICS region was in RLS mode.

SYSID(*data-area***)**
returns the 4-character sysid (when the CVDA on the CAUSE parameter is CONNECTION) of the remote system to which connectivity has been lost.

UOW(*data-area***)**
returns, as an 8-byte field, the UOW identifier of a shunted unit of work that has one or more data sets with retained locks.

Conditions

END
RESP2 values:

2 There are no more UOW/data set pairs.

ILLOGIC
RESP2 values:

1 A START has been given when a browse is already in progress, or a NEXT has been given without a preceding START.

NOTAUTH
RESP2 values:

100 The use of this command is not authorized.

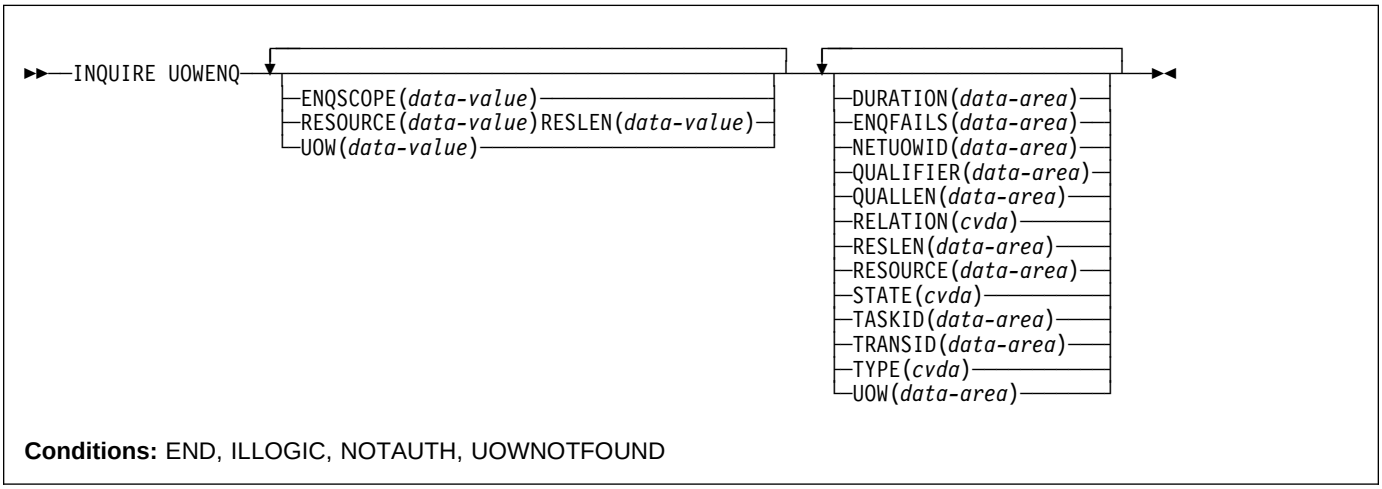
INQUIRE UOWENQ

Retrieve information about enqueues held or waited on by a UOW, or about UOWs holding or waiting on a specified enqueue.

INQUIRE ENQ is a synonym for INQUIRE UOWENQ.

Browse only function

The INQUIRE UOWENQ command can be used only in browse mode.



Conditions: END, ILLOGIC, NOTAUTH, UOWNOTFOUND

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

Description

The INQUIRE UOWENQ command retrieves information about enqueues. Enqueues are used by CICS to lock recoverable resources, such as file records or queues, to the UOW that is updating them. User enqueues obtained by the EXEC CICS ENQ command are also returned.

The browse can be filtered in three ways:

- Supply a value for UOW on the START command to return only the enqueues held or waited on by the specified UOW.
- Supply a value for RESOURCE on the START command to return only information about UOWs owning or waiting on the specified enqueue.
- Supply a value for ENQSCOPE on the START command to return only enqueues with the specified enqscope. If ENQSCOPE is specified as blanks, only local enqueues will be returned.

A CICS-wide browse occurs when you do not supply a value for UOW, RESOURCE or ENQSCOPE on the INQUIRE UOWENQ START command. All enqueue owners and enqueue waiters on the local system are returned by the browse. They are returned by considering each UOW in turn. After all the enqueues owned by one UOW have been returned, those owned by the next UOW in the system are considered.

As well as returning information about the owners of the enqueues, the command also returns information about UOWs that are waiting on these enqueues. This enables you to diagnose enqueue deadlocks between tasks wanting to update the same resources. It provides a performance improvement over other methods of answering the question "Which UOW is holding the Enqueue?" when you want to analyse what the cause of a delay is.

Enqueues are typically held in active state, which means that other tasks are allowed to wait for the enqueue. However, if a UOW that owns enqueues suffers an in-doubt failure, user ENQs are released while CICS enqueues are usually converted to the retained state until the in-doubt failure can be resolved. User ENQs are not to be used to lock recoverable resources, as they are not held across a CICS failure. The INQUIRE UOWENQ command also retrieves information about retained enqueues and can be used to identify which records and queues would be affected if the UOW were forced.

INQUIRE UOWENQ only returns information about UOWs on the local system. For Enqueues with SYSPLEX SCOPE the OWNER may be on the local system with some or all of the waiters elsewhere, or the enqueue OWNER may be elsewhere in the sysplex with some or all of the waiters on the local system; In this case, only the local waiters will be returned.

Browsing

Using the browse options (START, NEXT, and END) on INQUIRE UOWENQ commands, you can browse through all of the enqueues held by a specific UOW, or through all the enqueues currently in your system. See “Browsing resource definitions” on page 18 for general information about browsing, including syntax, exception conditions, and examples.

The browse returns both enqueue owners and enqueue waiters. They are returned by considering each UOW that owns an enqueue in turn. After all the enqueues owned by one UOW have been returned, those owned by the next UOW in the system are considered. Enqueue waiters are returned subsequent to the enqueue they are waiting on, but prior to the next enqueue owned by the current UOW. Note that the INQUIRE UOWENQ START does not retrieve data for the first enqueue. Also, because the enqueues are not returned in a defined order, you cannot specify a start point.

A CICS-wide browse occurs when you do not supply a value for UOW on the INQUIRE UOWENQ START command. All enqueue owners and waiters are returned by the browse. The first time an INQUIRE UOWENQ NEXT command is used, it returns the data for the first enqueue that is owned. This is returned with RELATION(OWNER). If the enqueue has any waiters, the same enqueue is returned for each of these waiters, but this time with RELATION(WAITER). The UOW, NETUOWID, TASKID, and TRANSID fields each correspond to that particular waiter. All other data should be the same as when it was returned with RELATION(OWNER). After the last waiter has been returned, the next time the command is issued it returns the next enqueue that is owned (if any).

If you supply a value for UOW on the START command, it acts as a “filter,” which means that only those enqueues owned by that particular UOW are returned (with a RELATION of OWNER). If the UOW happens to be waiting for an enqueue then this too is returned (but with a RELATION of WAITER).

Note that the enqueue state is not locked for the duration of the browse, or even between consecutive INQUIRE NEXT commands. To receive a consistent view of the state, the task performing the browse should not give up control to another task while the browse is in progress. If the owner of the last enqueue returned by the browse changes between successive INQUIRE NEXT commands, the browse returns the enqueue again with its new owner and waiters.

Notes:

1. If there are many enqueues in the system, CICS may take a long time to process a browse. If this happens, consider increasing the runaway interval of tasks that perform browses. (Do this by increasing the value of the RUNAWAY attribute on the associated TRANSACTION definition).
2. Both UOW-lifetime and task-lifetime enqueues are returned by INQUIRE UOWENQ. (For an explanation of UOW- and task-lifetime enqueues, see the MAXLIFETIME option of the EXEC CICS ENQ command.)
3. On an in-doubt failure, user enqueues are released, *unless* the EXEC CICS ENQ command specified MAXLIFETIME(TASK) and it is not the end-of-task syncpoint that suffers the failure.

Options

DURATION(*data-area***)**
returns, as a fullword value binary value, the elapsed time in seconds since the enqueue entered its current state of owner, waiter or retained.

ENQFAILS(*data-area***)**
returns, for retained enqueues, the number of failed enqueue attempts for this resource after the enqueue was last acquired. This indicates how many UOWs have received a LOCKED response because this enqueue was held in retained state. For active enqueues, ENQFAILS returns zero.

Because the ENQFAILS option indicates how many UOWs are failing because of retained locks, you can use it to help identify which shunted UOWs are causing “bottlenecks.”

ENQSCOPE(*data-area***)**
If the enqueue has sysplex scope, ENQSCOPE returns the 4-character name which was used to qualify the sysplex-wide ENQUEUE request issued by this CICS region. If it has region scope, ENQSCOPE returns blanks.

All CICS systems with the same ENQSCOPE value share the same sysplex Enqueue name space.

ENQSCOPE may also be used to supply a value on the START command. This limits the INQUIRE to return only enqueues with the specified scope name. If ENQSCOPE is specified as blanks, only local enqueues will be returned.

NETUOWID(*data-area***)**
returns the 1- through 27-character network-wide LU6.2 ID of the UOW that owns or is waiting for the enqueue for which data is being returned.

QUALIFIER(*data-area***)**
returns a 0- through 255-character optional qualifier that further identifies the resource associated with the enqueue. The data (if any) returned in this field depends on the TYPE of the enqueue, as summarized in Table 3 on page 225.

QUALLEN(*data-area***)**
returns a halfword binary value indicating the length of the data, in the range 0 through 255, returned in the QUALIFIER field. If no QUALIFIER data is applicable to

the resource (that is, for EXECQENQ, EXECENQADDR, and TSQUEUE), a value of zero is returned.

RELATION(*cvda***)**
returns a CVDA value indicating whether the data being returned is associated with the owner of the enqueue or with a task waiting for the enqueue. CVDA values are:

OWNER
The UOW, NETUOWID, TASKID, and TRANSID are those of the owner of the enqueue.

WAITER
The UOW, NETUOWID, TASKID, and TRANSID are those of a waiter for the enqueue.

RESLEN(*data-area***)**
returns a halfword binary value indicating the length of the data, in the range 1 through 255, returned in the RESOURCE field.

| If RESOURCE is used as input on a START command,
| a RESLEN input is also required.

RESOURCE(*data-area***)**
returns the 1- through 255-character name of the resource associated with the enqueue lock. The data returned in this field depends on the TYPE of the enqueue, as summarized in Table 3 on page 225.

| RESOURCE may also be used to supply a value on the
| START command. This limits the INQUIRE to return only
| information about UOWs owning or waiting on the
| specified enqueue.

STATE(*cvda***)**
returns a CVDA value indicating the state that the enqueue being returned is held in. It is returned on the INQUIRE UOWENQ NEXT command. CVDA values are:

ACTIVE
The enqueue is held in active state.

RETAINED
The enqueue is held in retained state. Its owning UOW has been shunted, or is in the process of being shunted.

TASKID(*data-area***)**
returns a 4-byte packed-decimal value giving the number of the task associated with the UOW. If the UOW is shunted, this is the task number associated with the UOW before it was shunted.

TRANSID(*data-area***)**
returns the 1- through 4-character identifier of the transaction associated with the UOW. If the UOW is shunted, it is the identifier of the transaction associated with the UOW before it was shunted.

TYPE(*cvda***)**
returns a CVDA value identifying the type of resource being enqueued upon. CVDA values are:

DATASET
The resource is a record in a VSAM RLS data set (or a CICS-maintained data table). Note that the data set is one open in RLS mode. CICS does not hold enqueues on non-RLS data sets, because locks are held by VSAM RLS for such data sets. RESOURCE contains the name of the data set and QUALIFIER contains the record identifier.

EXECENQ
The resource is associated with an EXEC CICS ENQ request. RESOURCE contains the enqueue argument passed on the request.

EXECENQADDR
The resource is associated with an EXEC CICS ENQ request. RESOURCE contains the address enqueue argument passed on the request (that is, the LENGTH parameter was omitted on the request)

FILE The resource is a record in either a BDAM file or a user-maintained data table. RESOURCE contains the name of the file and QUALIFIER contains the record identifier.

When the file is a BDAM file then the record identifier is prefixed by the BDAM block identifier. Note that truncation occurs if this combination exceeds 255 characters.

TDQUEUE
The resource is a logically-recoverable transient data queue. RESOURCE contains the name of the queue. QUALIFIER contains either the string “FROMQ” or “TOQ,” indicating whether an input or output lock is held for that queue.

Note that the definition of the WAITACTION attribute on the TDQUEUE resource definition determines what happens to TDQUEUE enqueues on an indoubt failure. For information on defining the WAITACTION attribute, see the *CICS Resource Definition Guide*.

A READQ TD request acquires the “FROMQ” lock, whereas a WRITEQ TD request acquires the “TOQ” lock associated with the queue. A DELETEQ TD request acquires both the “TOQ” and the “FROMQ” locks.

TSQUEUE
The resource is a recoverable temporary storage queue. RESOURCE contains the name of the queue.

Unlike other components, enqueues associated with recoverable temporary storage queues are only ever the retained kind; owned by a UOW that has been shunted as a result of an in-doubt failure. The temporary storage component uses its own mechanism for locking queues to in-flight UOWs.

The data returned in the RESOURCE and QUALIFIER fields depends on the resource TYPE, as shown in Table 3 on page 225.

Table 3. Data returned in RESOURCE and QUALIFIER

TYPE	RESOURCE	QUALIFIER
DATASET	Data set name	Record identifier
EXECENQ	EXEC enqueue argument	None
EXECENQADDR	Address of EXEC enqueue argument	None
FILE	File name	Record identifier
TDQUEUE	TD queue name	FROMQ or TOQ
TSQUEUE	TS queue name	None

UOW(*data-area***)**
returns the 16-character local identifier of the UOW that owns or is waiting for the enqueue for which data is being returned. The last eight bytes are always null.

| UOW may also be used to supply a value on the START
| command. This limits the INQUIRE to return only the
| enqueues held or waited on by the specified UOW.

Conditions

END
RESP2 values:

2 All enqueues have been retrieved.

ILLOGIC
RESP2 values:

1 For INQUIRE UOWENQ START, means that a browse of this resource type is already in progress. For INQUIRE UOWENQ NEXT and INQUIRE UOWENQ END, means that an INQUIRE UOWENQ START command has not been issued.

NOTAUTH
RESP2 values:

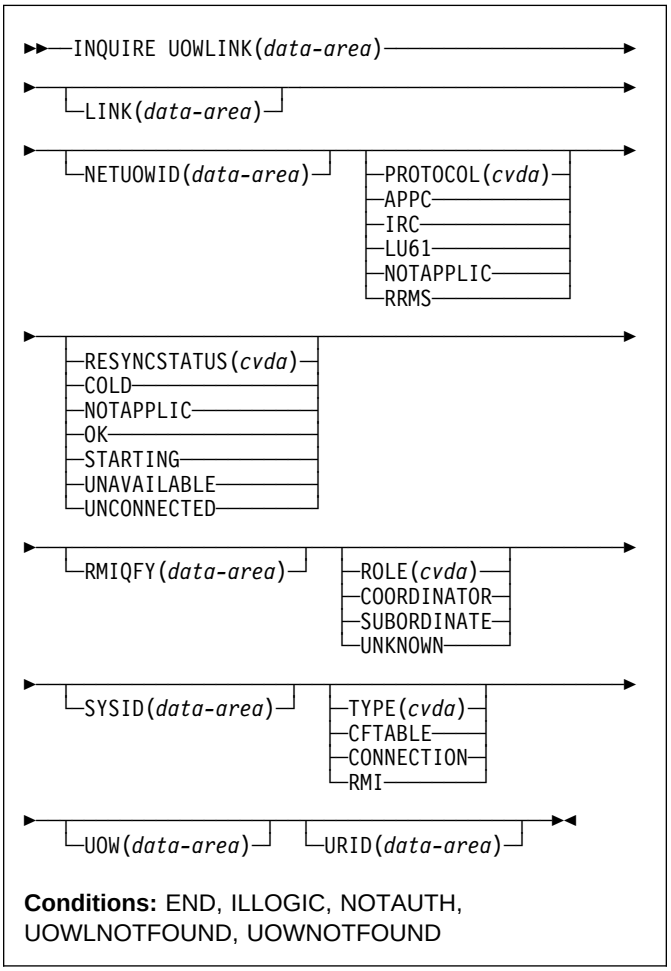
100 The use of this command is not authorized.

UOWNOTFOUND
RESP2 values:

1 The named UOW cannot be found.

INQUIRE UOWLINK

Retrieve information about a connection involved in a unit of work.



Conditions: END, ILLOGIC, NOTAUTH, UOWLNOTFOUND, UOWNOTFOUND

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

Description

The INQUIRE UOWLINK command retrieves information about a connection involved in a unit of work. The connection can be to a remote system, to a task-related user exit, or to a CFDT server.

If it is to a remote system, INQUIRE UOWLINK returns the netname of the connection, its sysid, and whether it is the coordinator or subordinate. If it is to a task-related user exit, INQUIRE UOWLINK returns the exit's entry name and qualifier.If it is to a CFDT server, INQUIRE UOWLINK returns the poolname.

Browsing

You can browse through all UOW-links by using the browse options (START, NEXT, and END) on INQUIRE UOWLINK commands. See "Browsing resource definitions" on page 18 for general information about browsing, including syntax, exception conditions, and examples.

The browse form of the command returns the state of all the UOW-links for connections that are *directly* connected to the CICS from which the command is issued. It indicates which connections are unavailable, or have been cold started.

The INQUIRE UOWLINK START command positions an internal pointer at the first UOW-link in the CICS recovery manager table. It does not retrieve data for the first one, and it does not allow you to specify a start point.

The first time an INQUIRE UOWLINK NEXT command is used, it returns information about the first UOW-link. Each time the command is used again, it retrieves the ID and STATE of the next UOW-link (if there is one). You can filter the UOW-links returned by specifying a value in the UOW field.

The browse guarantees that data for each UOW-link that exists prior to the first INQUIRE NEXT, and still exists after the last INQUIRE NEXT, is returned on exactly one INQUIRE NEXT call.

Options

LINK(*data-area***)**
returns, for a TYPE value of CONNECTION, the 8-character netname of the remote system. For a TYPE value of RMI, LINK returns the entry name of the task-related user exit. For a TYPE value of CFTABLE, LINK returns the 8–character name of the coupling facility data table pool.

NETUOWID(*data-area***)**
returns the 1-through 27-character network-wide LU6.2 ID of the UOW for which data is returned.

PROTOCOL(*cvda***)**
returns a CVDA value indicating the communication protocol used by the connection. CVDA values are:

APPC
Advanced Program to Program Communication.

IRC InterRegion Communication. This is an MRO connection.

LU61
LUTYPE 6.1.

NOTAPPLIC
This is a connection of type CFTABLE or RMI.

RRMS
The UOW is coordinated by RRS/MVS.

RESYNCSTATUS(*cvda***)**
returns a CVDA value indicating the resynchronization status of the connection. CVDA values are:

COLD
The connection has been cold started by the partner system. The partner can no longer coordinate any in-doubt conditions for this system; nor can this system pass to the partner any decisions remembered for it.

NOTAPPLIC
The connection was not created using recovery protocols. It could be an RMI, an APPC single-session, an APPC synclevel 1 connection, or a CFDT server.

OK The connection is operating normally. If there has been a failure of the partner system, the partner has been restarted and the connection is able to resynchronize the associated UOW.

STARTING
The connection is being acquired, but the exchange lognames process has not yet completed.

UNAVAILABLE
The connection is not currently acquired.

UNCONNECTED
There is no associated connection.

RMIQFY(*data-area***)**
returns, for a TYPE value of RMI, the 8-character entry qualifier of the task-related user exit. For a TYPE of CONNECTION, or CFTABLE, RMIQFY returns blanks.

ROLE(*cvda***)**
returns a CVDA value indicating the role of the connection. CVDA values are:

COORDINATOR
This connection is to the syncpoint coordinator for the UOW.

SUBORDINATE
This connection is to a syncpoint subordinate for the UOW.

UNKNOWN
The syncpoint role of this connection cannot be determined.

SYSID(*data area***)**
returns, for a TYPE value of CONNECTION, the 4-character sysid of the connection. If the connection has been discarded, or the type is RMS, or CFTABLE, or the PROTOCOL option returns RRMS, SYSID returns blanks.

TYPE(*cvda***)**
returns a CVDA value indicating the type of connection. CVDA values are:

CFTABLE
A connection to a CFDT server.

CONNECTION
A connection defined in a CONNECTION resource definition.

RMI A connection to an external resource manager using the resource manager interface (RMI).

UOW(*data-area***)**
returns the 16-byte local identifier of the UOW for which link data is being returned. The last eight bytes are always null.

UOWLINK(*data-area***)**
specifies a 4-byte token identifying the UOW-link for which data is to be returned.

URID(*data-area***)**
If the PROTOCOL field returns RRMS, this option returns the 32 byte hexadecimal representation of the RRMS unit of recovery identifier. For other values of PROTOCOL, URID returns blanks

Conditions

END
RESP2 values:

2 All authorized resource definitions have been retrieved.

ILLOGIC
RESP2 values:

1 For INQUIRE UOWLINK START, means that a browse of this resource type is already in progress. For INQUIRE UOWLINK NEXT and INQUIRE UOWLINK END, means that an INQUIRE UOWLINK START command has not been issued.

NOTAUTH
RESP2 values:

100 The use of this command is not authorized.

UOWLNOTFOUND
RESP2 values:

1 The named UOW-link cannot be found.

UOWNOTFOUND
RESP2 values:

1 The named UOW cannot be found.

INQUIRE VOLUME

This command is supported in releases of CICS earlier than CICS Transaction Server for OS/390, for retrieving information about journal volumes.

Description

INQUIRE VOLUME is obsolete, and is retained only for compatibility with previous releases. The only run-time support is to return the VOLIDERR condition. If this command is used, the translator translates it, but issues a warning message.

The browse function is provided for compatibility with releases of CICS earlier than CICS Transaction Server for OS/390. A NORMAL condition is returned for the START browse and END browse operations. The ENDCOND condition is returned for the NEXT browse operation.

Conditions

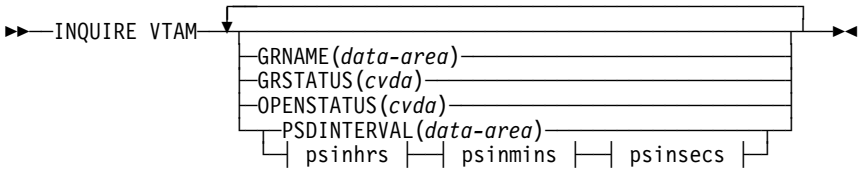
VOLIDERR

RESP2 values:

1 The program has issued an INQUIRE VOLUME browse command. This command is withdrawn.

INQUIRE VTAM

Retrieve information about the connection between CICS and VTAM.



psinhrs:
PSDINTHRS(*data-area*)

psinmins:
PSDINTMINS(*data-area*)

psinsecs:
PSDINTSECS(*data-area*)

Conditions: INVREQ, NOTAUTH

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 reregister.

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 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 are not 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:

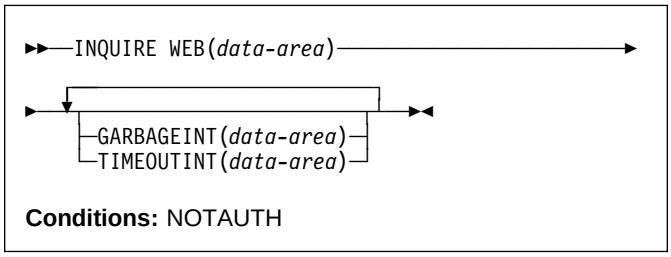
100 The user associated with the issuing task is not authorized to use this command.

INQUIRE WEB

Retrieve information about CICS Web support.

Context

INQUIRE WEB returns information about the state of CICS Web support.



Conditions: NOTAUTH

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

Description

The INQUIRE WEB command returns information about the status of the CICS Web interface.

Options

GARBAGEINT(*data-area***)**
returns, in fullword binary form, the interval, in seconds, at which the Web garbage collection task runs to clean up Web 3270 state data for which the terminal timeout interval has expired.

TIMEOUTINT(*data-area***)**
returns, in fullword binary form, the period of time, in seconds, after which inactive Web 3270 sessions are eligible for garbage collection.

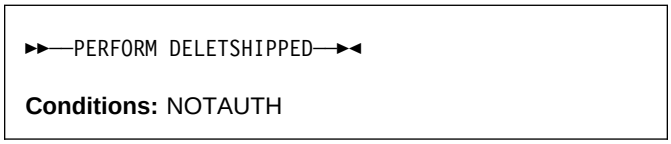
Conditions

NOTAUTH
RESP2 values:

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

PERFORM DELETSHIPPED

Delete inactive shipped terminal definitions.



Description

The PERFORM DELETSHIPPED 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 DELETSHIPPED command, and you can set these values with the SET DELETSHIPPED command. For more information about shipped definitions, see the *CICS Intercommunication Guide* and the *CICS Resource Definition Guide*.

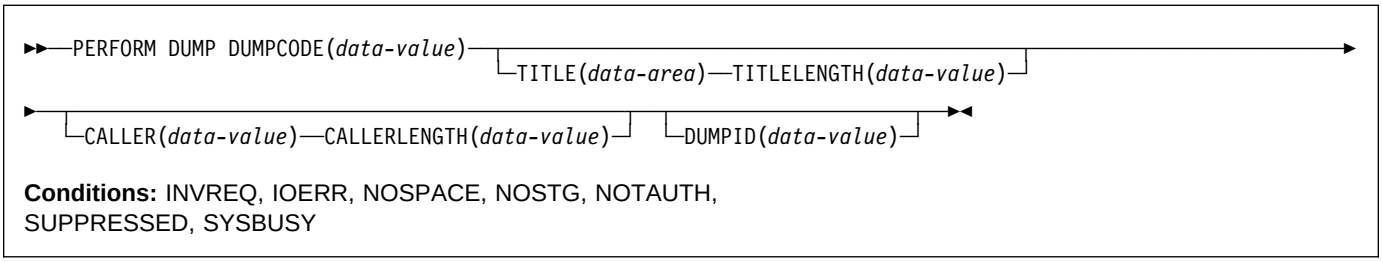
Conditions

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.



Conditions: INVREQ, IOERR, NOSPACE, NOSTG, NOTAUTH, SUPPRESSED, SYSBUSY

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 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 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.

TITLE(*data-area***)**
is 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:

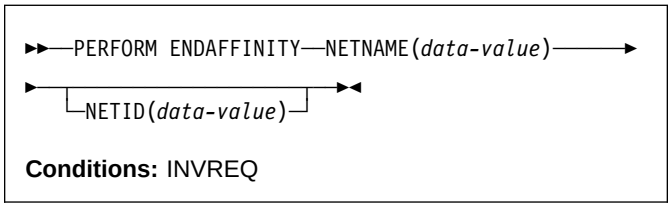
1 The dump was not taken because the number of dumps with this dump code exceeds the maximum for the code.
2 The dump was not taken because the system dump table entry for this code indicates no system dump.
3 The dump was not taken because it was suppressed by a user exit program.
8 The dump was not taken because system dumps are suppressed globally.

SYSBUSY
RESP2 values:

11 The MVS dump routine is busy. Retry the command.

PERFORM ENDAFFINITY

End an affinity owned by CICS.



Conditions: INVREQ

Description

Where CICS is a member of a VTAM generic resource group, the PERFORM ENDAFFINITY command instructs VTAM to end an affinity owned by CICS, whether or not the connection has been deleted. If the connection has not been deleted, it must be out of service and have no recovery information outstanding (that is, its RECOVSTATUS must be NORECOVDATA).

Generic resources and affinities are described in the *CICS Intercommunication Guide*.

Note: There is no facility in VTAM for inquiring on affinities, so CICS has no certain knowledge that an affinity exists for a given connection. Whenever there is a possibility that an affinity has been created that you must end explicitly, CICS issues message DFHZC0177. This message gives the NETNAME and NETID to be passed to VTAM.

If a request to end an affinity is rejected by VTAM because no such affinity exists, CICS issues message DFHZC0181. This may mean either that your program specified an incorrect NETNAME or NETID, or that it (or CICS) was wrong in supposing that an affinity existed.

Options

NETID(*data-value***)**
specifies the name by which the network containing the remote LU is known to VTAM.

If you do not specify a NETID, CICS takes the value from the installed connection, if it exists. If you do not specify a NETID and the connection does not exist, the command fails.

NETNAME(*data-value***)**
specifies the APPLID of the remote LU. If the connected LU is a member of a generic resource, you must specify its member name, not the generic resource name.

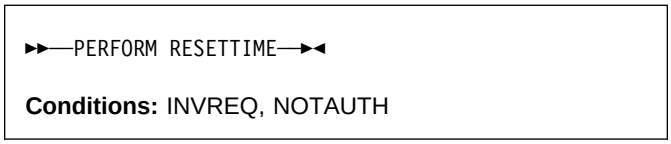
Conditions

INVREQ
RESP2 values:

25 The connection is still in service.
26 There may be recovery information outstanding for the connection. RECOVSTATUS has a value other than NORECOVDATA.
32 See message DFHZC0178. VTAM could not end the affinity for a reason other than 35 (NOTFOUND) or 36 (SESSIONS ACTIVE).
34 NETID was not specified, and cannot be obtained from the installed connection. This may be because the connection does not exist, or because it does not contain a NETID value.
35 VTAM could not find an affinity for the values input.
36 VTAM could not end the affinity because the connection had some sessions active.
37 See message DFHZC0176. A VTAM error prevented the CHANGE ENDAFFIN macro being carried out.

PERFORM RESETTIME

Reset date and time.



Description

The PERFORM RESETTIME command resets the CICS date and time from the MVS system date and time.

Conditions

INVREQ
RESP2 values:

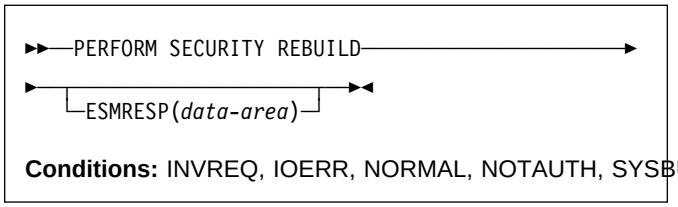
1 There is no clock in the system.

NOTAUTH
RESP2 values:

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

PERFORM SECURITY REBUILD

Refresh security information.



Conditions: INVREQ, IOERR, NORMAL, 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.

Since 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.

If the ESM is not RACF Version 2.1 or later, 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 later, 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.

NORMAL
RESP2 values:

0 Profiles have been refreshed.
4 No action is required, because RACF is at Version 2.1 (or later), and so profiles are refreshed automatically.

NOTAUTH
RESP2 values:

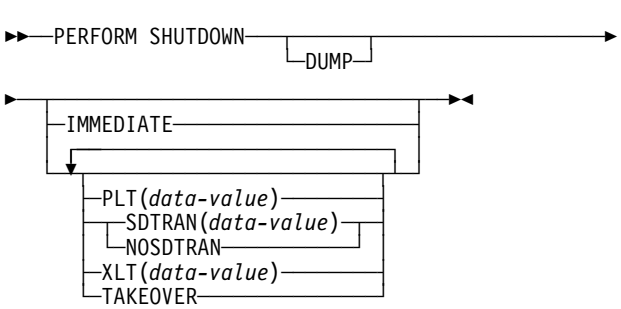
100 The user associated with the issuing task is not authorized to use this command.

SYSBUSY
RESP2 values:

3 A security rebuild is currently in progress.

PERFORM SHUTDOWN

Shut down the CICS system.



Conditions: INVREQ, NOTAUTH, TRANSIDERR

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 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; it then takes a warm keypoint.

NOSDTRAN
specifies that no shutdown assist transaction is to be run at CICS shutdown.

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.

SDTRAN(*data-value***)**
specifies the 4-character name of the shutdown assist transaction.

The shutdown assist transaction, if specified, is run at CICS warm and immediate shutdown, and can be used to ensure that CICS shuts down in an controlled way, within a reasonable time (by, for example, purging long-running tasks). For details of the default shutdown assist transaction, CESD, see the *CICS Operations and Utilities Guide*.

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

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.

5 The transaction specified on SDTRAN is not enabled for shutdown.
6 The transaction specified on SDTRAN is defined as remote.
7 The transaction specified on SDTRAN is not enabled.

NOTAUTH
RESP2 values:

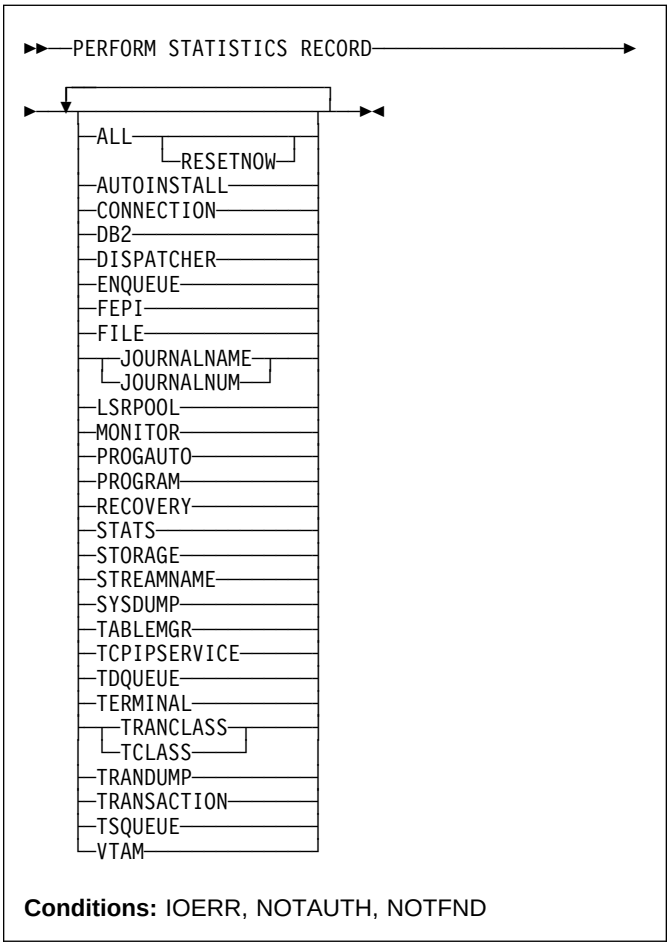
100 The user associated with the issuing task is not authorized to use this command.

TRANSIDERR
RESP2 values:

8 The shutdown transaction specified on SDTRAN was not found.

PERFORM STATISTICS RECORD

Record statistics immediately.



Conditions: IOERR, NOTAUTH, NOTFND

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 286.)

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 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.

DB2
records global statistics for the CICS DB2 connection and specific statistics for each DB2ENTRY defined in your system.

DISPATCHER
records global statistics on the dispatch function, including task counts and concurrency levels and limits.

ENQUEUE
records global statistics for the enqueue manager.

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 installed in your system.

JOURNALNAME
records specific statistics for all journals installed in your system. This parameter replaces the JOURNALNUM parameter. To record specific statistics for all journals installed in your system, you are recommended to use this parameter.

JOURNALNUM
records specific statistics returned by the JOURNALNAME parameter.

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.

RECOVERY
records global statistics on the recovery manager.

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 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.

STREAMNAME
records specific statistics for all the log streams currently connected.

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.

TCPIPSERVICE
records specific statstics for every TCP/IP service installed in your system.

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.

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:

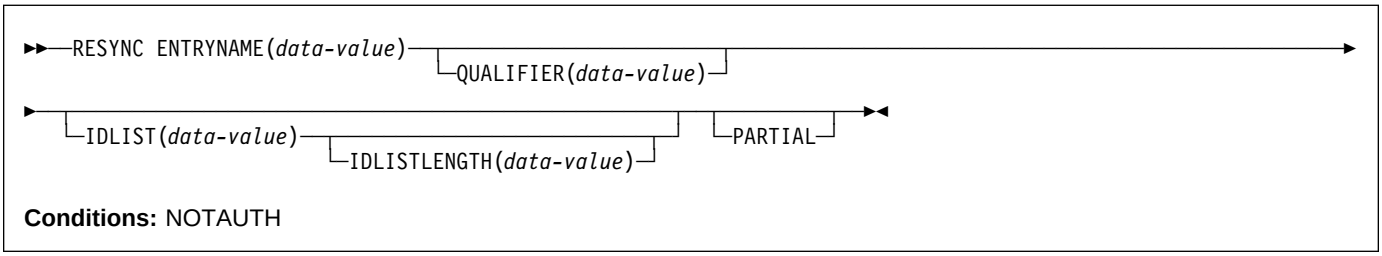
n Statistics for at least one of the options chosen were not available because CICS was initialized without support for the function. (See note below.)

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
6	FILE
8	JOURNALNUM and JOURNALNAME
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
28	ENQUEUE
29	RECOVERY
30	STREAMNAME
31	DB2
32	TCPIPSERVICE

RESYNC ENTRYNAME

Determine the disposition of “in doubt” units of work.



Conditions: NOTAUTH

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. 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 an initial 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 program or an associated administrative transaction.

In response, CICS creates a task, CRSY, for each in-doubt unit of work in the list. The CRSY task invokes the task-related user exit program 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 program can then relay this information in the form the resource manager requires.

If the resource manager does not want to resynchronize all in-doubt units of work at once, it should specify PARTIAL on the RESYNC command. If it does not, CICS discards disposition information for all the in-doubt units of work that are not in the supplied list, but are part of the resource manager’s resynchronization set. [3]

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 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

[3] A resource manager’s resynchronization set is initialized when its task-related user exit is first enabled. It is used when the first non-partial RESYNC command is issued. On completion of the non-partial RESYNC, a new resynchronization set is initialized, for use with the next non-partial RESYNC.

for the resource manager, unless you specify PARTIAL. Not specifying a list and specifying PARTIAL is an illogical combination and results in a NO-OP.

IDLISTLENGTH(*data-value***)**
specifies a halfword binary value indicating the length (in bytes, counting 4 bytes per in-doubt unit of work) of the address-list.

PARTIAL
specifies that CICS is to retain in-doubt resolution data for the UOWs (for this resource manager) that are not passed in the in-doubt list. PARTIAL indicates that, at this time, the resource manager wants to resynchronize only a subset of the UOWs about which it is in doubt.

If PARTIAL is not specified, CICS discards resolution data for any UOWs not passed in the in-doubt list, but which are part of this resource manager's resynchronization set. [3] This includes data for UOWs that CICS itself is in doubt about.

A task-related user exit program can issue multiple partial resyncs during the lifetime of a connection with its external resource manager. However, it should issue only *one* full (that is, non-partial) resync during the lifetime of a connection. This is typically done when the connection is first established. Full resyncs imply deletion of UOWs not mentioned in the IDLIST. Only when the external resource manager is not connected to CICS can it be sure that it has a complete list of UOWs to pass to CICS.

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 with the same QUALIFIER and ENTRYNAME values. Similarly, it discards saved dispositions only if they have the same two values, were not included in the IDLIST, and PARTIAL was not specified.

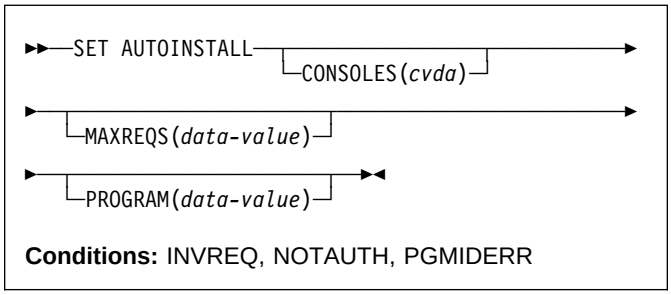
Conditions

NOTAUTH
RESP2 values:

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

SET AUTOINSTALL

Change autoinstall values.



Conditions: INVREQ, NOTAUTH, PGMIDERR

Description

The SET AUTOINSTALL command lets you change some of the values that control the automatic installation (autoinstall) of VTAM terminals, APPC sessions, and MVS consoles in a CICS region.

Options

CONSOLES(*cvda***)**
specifies whether CICS is to autoinstall an MVS console when it receives an MVS MODIFY command from a console that is not defined. The CVDA values are:

PROGAUTO
MVS consoles are to be autoinstalled, and CICS is to call the user autoinstall control program to obtain the termid and other user-specified information.

FULLAUTO
MVS consoles are to be autoinstalled by CICS automatically, without calling the user autoinstall control program. CICS assigns the termid for the console automatically, using the ¬ (logical not) symbol as the first character.

NOAUTO
Autoinstall for consoles is not allowed.

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 program autoinstall process. You can do this either with explicit PROGRAM definitions or by autoinstall when some other autoinstall program is in force. Otherwise, the program autoinstall process fails when it is next used, and CICS makes it inactive.

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.
20 The console has an invalid CVDA value.

NOTAUTH
RESP2 values:

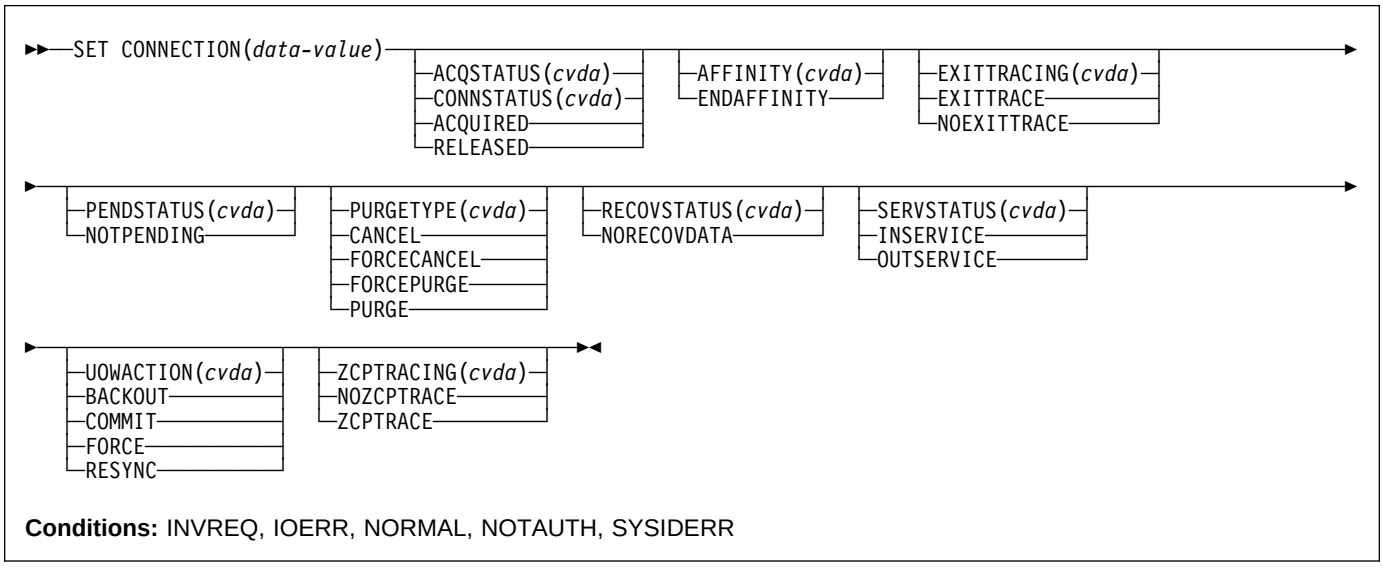
100 The user associated with the issuing task is not authorized to use this command.

PGMIDERR
RESP2 values:

3 The program name cannot be found.

SET CONNECTION

Change some connection attributes and cancel outstanding AIDs.



Conditions: INVREQ, IOERR, NORMAL, NOTAUTH, SYSIDERR

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

Description

The SET CONNECTION command allows you to change some of the attributes that define a connection. Control returns to the issuing program when the required operation has been scheduled. To get the operation started, it is necessary to relinquish control to CICS.

There are two main types of system connection:

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

  A special form of IRC 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.

The process of acquiring and releasing APPC sessions involves starting the LU Services Manager transaction CLS1. In order to pass data to the CLS1 transaction, CICS uses a temporary storage queue with the default data identifier (dataid) prefix DF. If temporary storage dataids with the prefix DF are defined as recoverable in your installation, you must follow the SET CONNECTION command by a SYNCPOINT command to end the logical unit of work and allow the SET CONNECTION command to complete.

Options

ACQSTATUS(*cvda***) (APPC only)**
is retained only for compatibility purposes. You should use CONNSTATUS in new applications.

AFFINITY(*cvda***) (APPC and LU61 only)**
specifies, where CICS is a member of a VTAM generic resource group, that VTAM is to end an affinity owned by CICS. This option is valid only for APPC and LU6.1 connections. The connection must be out of service and, for APPC, in NORECOVDATA state.

The CVDA value is:

ENDAFFINITY
End the affinity.

Notes:

1. There is no facility in VTAM for inquiring on affinities, so CICS has no certain knowledge that an affinity exists for a given connection. Whenever there is a possibility that an affinity has been created that must be ended explicitly, CICS issues message DFHZC0177. This message gives the NETNAME and NETID of the suspect connection.
2. If a request to end an affinity is rejected by VTAM because no such affinity exists, CICS issues message DFHZC0181.

3. Generic resources and affinities are described in the *CICS Intercommunication Guide*.

CONNECTION(*data-value***)**
specifies, as a 4-character field, the APPC, IRC, or LUTYPE6.1 connection. This is the name of the remote system or region specified in the CONNECTION option of the CEDA DEFINE CONNECTION command.

CONNSTATUS(*cvda***) (APPC only)**
specifies whether to acquire or release sessions with the logical unit represented by the CONNECTION name. To get more detailed information about the availability status of the connection elements, use the INQUIRE MODENAME START, NEXT, and END commands. A connection cannot be both ACQUIRED and OUTSERVICE.

CVDA values are:

ACQUIRED
Sessions are to be acquired.

RELEASED
Sessions are to be released.

For further information about managing APPC connections, see the *CICS Intercommunication Guide*.

Note: CONNSTATUS is applicable to IRC connections for the INQUIRE CONNECTION command but not for the SET CONNECTION command. The CONNSTATUS of an MRO connection is controlled by setting the connection INSERVICE or OUTSERVICE using the SERVSTATUS CVDA.

EXITTRACING(*cvda***) (VTAM only)**
specifies whether to trace the activity associated with the terminal exit program for the sessions associated with this connection. CVDA values are:

EXITTRACE
The activity is to be traced.

NOEXITTRACE
The activity is not to be traced.

PENDSTATUS(*cvda***)(APPC and CICS-to-CICS MRO only)**
specifies, for either of the following kinds of connection, that the normal resynchronization process is to be overridden:

- A connection to a CICS Transaction Server for OS/390 partner that has performed an initial start
- A connection to a pre-CICS Transaction Server for OS/390 partner that has performed a cold start.

The CVDA value is:

NOTPENDING
Forces all in-doubt units of work (according to the transaction definition) that were created by the connection before the initial (or cold) start of the partner. It also forgets any resyncs (waitforget UOW-links) that are outstanding for the connection, and created before the initial (or cold) start of the partner.

The PENDING condition indicates the existence of recovery information (either shunted UOWs or decisions remembered for the partner) on a connection that has experienced a lognames mismatch with its partner. For a CICS Transaction Server for OS/390 partner, a lognames mismatch indicates that the partner has performed an initial start. For a pre-CICS Transaction Server for OS/390 partner, a lognames mismatch indicates that the partner has performed a cold start. In either case, the recovery protocol has been corrupted by a loss of log data at the partner.

It is not possible to set a connection to NOTPENDING state (forcing in-doubt and erasing NOFORGET UOWs) until this system has made contact with the partner and received a new logname from it.

Decisions for a whole connection can be forgotten, but that does not affect the memory of a decision for any other connection involved in the UOW.

Notes:

1. SET CONNECTION NOTPENDING, SET CONNECTION NORECOVDATA, and SET CONNECTION UOWACTION are mutually exclusive. For advice on which command to use, see the description of the UOWACTION option on page 245.
2. NOTPENDING has no effect on MRO connections to pre-CICS Transaction Server for OS/390 systems. However, the cold start of a pre-CICS Transaction Server for OS/390 MRO partner causes the SET CONNECTION NOTPENDING function to be performed automatically, session by session.

The exchange lognames function and the resynchronization function are described in the *CICS Intercommunication Guide* and the *Systems Network Architecture—LU6.2 Reference: Peer Protocols* manual.

PURGETYPE(*cvda***)**
specifies how associated transactions are to be purged. CVDA values are:

CANCEL
AIDs queuing for the specified connection are to be canceled.

AIDs representing scheduled and allocated requests waiting in the local CICS system for the specified connection are canceled. However, TD AIDs with an associated triggered task already started are not canceled. In addition, the following CICS system AIDs are not purged unless FORCECANCEL is specified.

Table 4. System AIDs requiring FORCECANCEL to remove them	
Remote delete AIDs	
Remote scheduler AIDs	CRSR
LU6.2 service manager 1 AIDs	CLS1
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

Message DFHTF0101 is written to CSMT to indicate how many AIDs have been deleted for the connection and how many remain.

When a canceled SCHEDULE request is found to have a precursor in a remote CICS system; that is, the AID was originally scheduled in a remote system, this remote AID is canceled asynchronously.

FORCECANCEL
: All AIDs, including system AIDs, queuing for the specified connection are to be canceled. See Table 4 for a list of those system AIDS that require FORCECANCEL to remove them. This can lead to unpredictable results and should be used only in exceptional circumstances.

  Note: FORCECANCEL does not remove transient data AIDs with an associated triggered task. These aids may be removed by purging the associated task.

FORCEPURGE
: All transactions running on sessions on the connected system are immediately terminated abnormally. This can lead to unpredictable results and should be used only in exceptional circumstances.

  In some extreme cases (for example, if an error occurs during backout processing), CICS might terminate abnormally.

  For in-doubt and shunted UOWs, FORCEPURGE has no effect.

  Note: To force shunted UOWs, the operator must issue SET CONNECTION COMMIT, BACKOUT, or FORCE following a FORCEPURGE. This can lead to unpredictable results and should be used only in exceptional circumstances.

PURGE
: Transactions running on the connected system are abnormally terminated. Transactions are terminated only if system and data integrity can be maintained. A transaction is not purged if its definition specifies SPURGE=NO, or if the UOW is shunted.

RECOVSTATUS(*cvda*) (APPC only)
: specifies that the normal resynchronization process is to be overridden. The CVDA value is:

  NORECOVDATA
  : Forces all in-doubt units of work (according to the transaction definitions), targets any resyncs that were outstanding for the connection, and erases the logname previously received from the partner system. The state of the connection is reset.

    Attention: You should use SET CONNECTION NORECOVDATA only in exceptional circumstances. It erases recovery information and may compromise data integrity for units of work that have updated resources on remote systems.

    Examples of circumstances in which you might need to use it are:

    - You need to discard a connection, or issue a SET CONNECTION ENDAFFINITY command, and it is not possible for the quiesce protocols with the partner system to be completed. (Neither action is possible for an APPC connection if recovery data is outstanding.)
    - An operational or logic error results in a logname mismatch for the connection. The connection state must be reset to allow the exchange lognames process to complete.

    Note: SET CONNECTION NORECOVDATA, SET CONNECTION NOTPENDING, and SET CONNECTION UOWACTION are mutually exclusive.

SERVSTATUS(*cvda*)
: specifies whether the system is to be placed in service or out of service. CVDA values are:

  INSERVICE
  : The system is to be placed in service; that is, to be available for use.

    For an MRO connection, all sessions are placed in service and the following occurs:

    - If both the issuing system and the remote system have IRC open, and the remote system has INSERVICE connection definition for the issuing system, the connection is made ACQUIRED (see the note following the description of the CONNSTATUS option).
    - Otherwise, the status of the connection is set INSERVICE so that the connection will be acquired when the above conditions are met.
    - The status of the underlying sessions for a connection is always the same as that for the connection itself.

For an EXCI connection, all receive sessions (or "pipes") are placed in service and available for use by the client program.

For an ISC APPC connection, the LU Services Manager sessions are placed in service, thereby enabling the connection subsequently to be acquired.

For an ISC LU6.1 connection, all sessions are placed in service.

OUTSERVICE
The connection is to be placed out of service; that is, not available for use.

For a connection, all sessions are placed out of service (immediately if PURGE is specified, or when tasks have terminated if it is not) and the following occurs:

- If the connection is currently ACQUIRED, the sessions are broken (quiesced). The connection cannot be used until it is once again placed INSERVICE.
- If the connection is currently RELEASED, the status of the connection is set OUTSERVICE and it cannot be used until it is INSERVICE again.
- The status of the underlying sessions for a connection is always the same as that for the connection itself.

For an EXCI connection, all receive sessions (or "pipes") are placed out of service and are not available for use by the client program.

For an ISC APPC system, this option is valid only if the connection is RELEASED. The LU Services Manager sessions are placed out of service, and the connection cannot be ACQUIRED until it is placed INSERVICE again.

For an ISC LU6.1 connection, all sessions are released and placed out of service: immediately if PURGE or FORCEPURGE is specified; or when tasks have terminated if neither PURGE nor FORCEPURGE is specified. If the response to an INQUIRE CONNECTION command shows OUTSERVICE, it does not imply that the connection has been explicitly set as SET OUTSERVICE; in particular circumstances, you cannot reinstall this connection.

UOWACTION(*cvda*) (APPC parallel-session, CICS-to-CICS MRO, and LU61 only)
specifies that the normal resynchronization process is to be partially overridden: decisions are taken for any units of work that are in-doubt because of a failure of the connection; but the decisions are recorded and any data inconsistencies are reported when the connection is next acquired.

The operation is synchronous with setting the state of the UOW; that is, an INQUIRE UOW following a SET CONNECTION UOWACTION returns the new UOW states. CVDA values are:

BACKOUT
All UOWs shunted because of the failure of this connection are to be backed out.

COMMIT
All UOWs shunted because of the failure of this connection are to be committed.

FORCE
All UOWs shunted because of the failure of this connection are to be forced to BACKOUT or COMMIT, as specified on the ACTION option of the TRANSACTION definition.

RESYNC (MRO-to-CICS Transaction Server for OS/390 and later systems, and APPC only)
Any UOWs shunted because of the failure of this connection are to be retried (that is, exchange lognames resynchronization for this connection is to be attempted). This process should normally be started automatically when a connection is acquired or when a UOW is unshunted.

Notes:

1. SET CONNECTION UOWACTION unshunts all units of work that have failed in-doubt because of a failure of the connection. Before issuing SET CONNECTION FORCE, you may want to use the SET UOW command to specify commit or backout for each in-doubt unit of work explicitly, rather than letting it default. Local procedures will determine the importance of the data and the method of using the INQUIRE UOW, INQUIRE UOWENQ, and INQUIRE UOWLINK commands to establish the correct actions.
2. As far as shunted units of work are concerned, you may use only one of SET CONNECTION UOWACTION, SET CONNECTION NOTPENDING, and SET CONNECTION NORECOVDATA. SET CONNECTION NORECOVDATA should be used only in exceptional circumstances.
3. To force all in-doubt units of work caused by a failure of the connection in the same direction, use SET CONNECTION COMMIT or SET CONNECTION BACKOUT.
4. Neither SET CONNECTION UOWACTION nor the SET UOW UOWACTION command clears resync information. If you want to do this, you must use SET CONNECTION NOTPENDING or SET CONNECTION NORECOVDATA.
5. You can issue SET UOW UOWACTION commands *before* issuing SET CONNECTION NOTPENDING or SET CONNECTION NORECOVDATA.

ZCPTRACING(*cvda*) (VTAM only)
specifies whether the VTAM control component of CICS is to trace activity on the sessions associated with this connection. CVDA values are:

NOZCPTRACE
VTAM ZCP tracing is not to be carried out.

ZCPTRACE
VTAM ZCP tracing is to be carried out.

Conditions

INVREQ
RESP2 values:

1 ACQSTATUS|CONNSTATUS specified for a non-APPC connection.

2 ACQUIRED and OUTSERVICE are specified inconsistently in any of the following ways:

1. ACQUIRED specified with OUTSERVICE
2. ACQUIRED specified for OUTSERVICE connection
3. OUTSERVICE specified for ACQUIRED connection.

3 ACQSTATUS|CONNSTATUS has an invalid CVDA value.

4 SERVSTATUS has an invalid CVDA value.

5 PENDSTATUS or NOTPENDING was specified for a connection that is not APPC or IRC.

6 PURGE was specified for a connection that is not VTAM.

7 PURGETYPE has an invalid CVDA value.

8 PENDSTATUS has an invalid CVDA value.

11 SET command named a remote connection.

12 EXITTRACING has an invalid CVDA value.

13 ZCPTRACING has an invalid CVDA value.

14 EXITTRACING|ZCPTRACING specified for a non-VTAM connection or VTAM not initialized.

16 The resource whose name was specified by CONNECTION(*data-value*) is an indirect link.

17 ACQSTATUS|CONNSTATUS cannot be set when system initialized with ISC=NO.

18 NOTPENDING cannot be set for a connection which has successfully completed Exchange Lognames processing.

19 CONNSTATUS cannot be set to ACQUIRED when in the FREEING state.

20 COMMIT, BACKOUT, FORCE, or RESYNC is not valid for this type of connection.

21 BACKOUT or FORCE was specified, but was unsuccessful. Some UOWs remain shunted for this connection.

22 Other SET parameters were included with the CANCEL or FORCECANCEL option.

23 The resource whose name was specified by CONNECTION(name) is the local TCT system entry (TCTSE).

25 Connection is still in service.

26 RECOVSTATUS does not have a value of NORECOVDATA.

30 Wrong connection type for ENDAFFINITY. Affinities can exist only on LU6.1 and LU6.2 connections.

31 The NETID could not be obtained from the installed connection. Therefore, to end an affinity you must use the PERFORM ENDAFFINITY command.

32 See message DFHZC0178. VTAM could not end the affinity for a reason other than 35 (NOTFOUND) or 36 (SESSIONS ACTIVE).

35 VTAM could not find an affinity for this connection.

36 VTAM could not end the affinity because the connection had some sessions active.

37 See message DFHZC0176. A VTAM error prevented the CHANGE ENDAFFIN macro being carried out.

44 GRSTATUS is not set to REGISTERED or DEREGISTERED. (No generic resource name.)

45 NORECOVDATA cannot be set for a connection that is in service.

46 NORECOVDATA was specified for a non-APPC connection.

IOERR
RESP2 values:

10 Unexpected error.

NORMAL
RESP2 values:

58 AIDs are successfully canceled.
59 No AIDs are canceled.

NOTAUTH
RESP2 values:

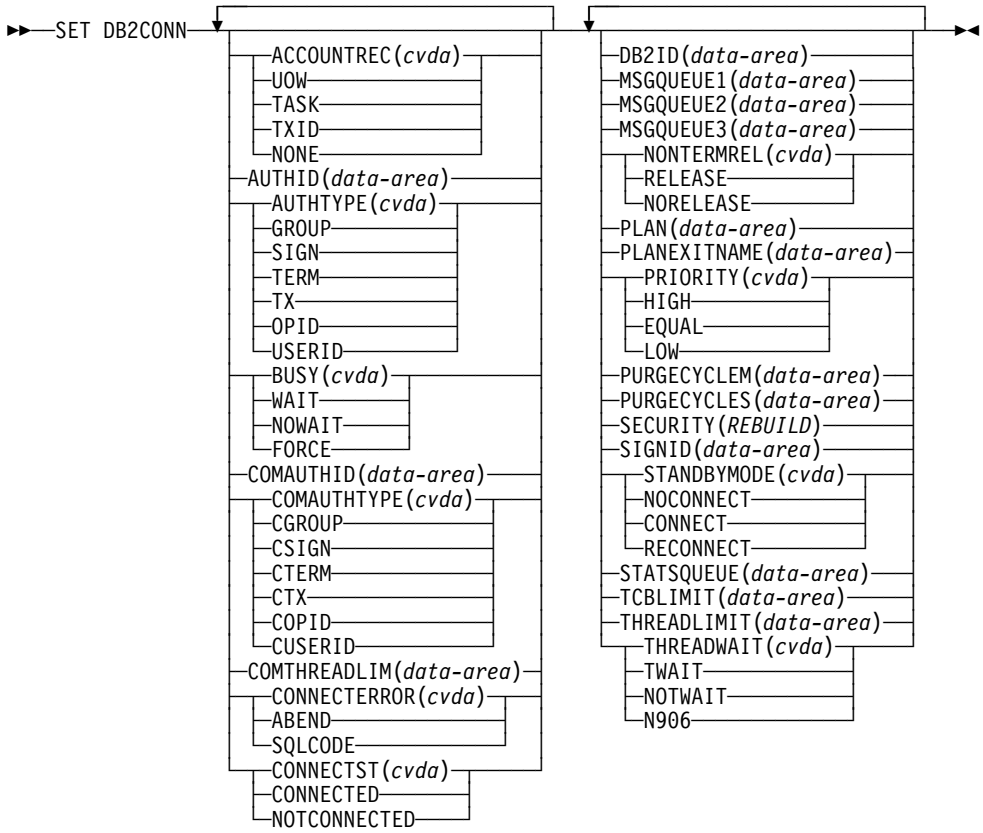
100 The user associated with the issuing task is not authorized to use this command.

SYSIDERR
RESP2 values:

9 The named connection could not be found.

SET DB2CONN

Change information about the attributes of the CICS DB2 Connection.



Conditions: NORMAL, NOTAUTH, NOTFND, INVREQ

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

Description

The SET DB2CONN command also specifies the attributes of the pool and command threads.

Options

ACCOUNTREC

Specifies the minimum amount of DB2 accounting required for transactions using pool threads. The specified minimum may be exceeded as described in the following options. CVDA values are:

NONE

No accounting records are required for transactions using pool threads.

DB2 produces at least one accounting record for each thread when the thread is terminated. Authorization changes additionally cause accounting records to be produced.

TXID

The CICS DB2 attachment facility causes an accounting record to be produced when the transid using the thread changes.

Because pool threads are typically used by a number of different transaction IDs, there is an increased chance that a transaction containing multiple units of work (UOWs) will use a different thread for each UOW (assuming the thread is released at syncpoint). In this case an accounting record may be produced per UOW.

TASK

The CICS DB2 attachment facility causes a minimum of one accounting record for each CICS task to be produced.

A transaction containing multiple UOWs (assuming the thread is released at syncpoint) may use a different thread for each of its UOWs. The result may be an accounting record produced for each UOW.

UOW
The CICS DB2 attachment facility causes an accounting record to be produced for each UOW, assuming that the thread is released at the end of the UOW.

AUTHID
specifies what id should be used for security checking for pool threads. If AUTHID is specified, AUTHType may not be specified.

AUTHTYPE
specifies the type of ID that can be used for pool threads. If AUTHType is specified AUTHID may not be specified. CVDA values are:

GROUP
Specifies the 8-character USERID and the connected group name as the authorization ID. The following table shows how these two values are interpreted by DB2.

	IDs passed to DB2	How DB2 interprets values
	CICS sign-on user ID (USERID)	Represents the primary DB2 authorization ID.
	RACF-connected group name	If the RACF list of group options is not active, DB2 uses the connected group name supplied by the CICS attachment facility as the secondary DB2 authorization ID. If the RACF list of group options is active, DB2 ignores the connected group name supplied by the CICS attachment facility, but the value appears in the DB2 list of secondary DB2 authorization IDs.

To use the GROUP option the CICS system must have SEC=YES specified in the CICS system initialization table (SIT).

If no RACF group ID is available for this USERID, an 8-character field of blanks is passed to DB2 as the group ID.

SIGN
Specifies the SIGNID parameter of the DB2CONN should be used as the resource authorization ID.

TERM
Specifies the terminal identification (four characters padded to eight) as an authorization ID. An authorization ID cannot be obtained in this manner if a terminal is not connected with the transaction.

If a transaction is started (using a CICS command) and has no terminal associated with it, AUTHTYPE(TERM) should not be used.

TX Specifies the transaction identification (four characters padded to eight) as the authorization ID.

OPID
The operator identification associated with the userid that is associated with the CICS transaction is used as the authorization ID (three characters padded to eight).

USERID
The 8-character USERID associated with the CICS transaction is used as the authorization ID.

When the DB2 sample sign-on exit DSN3@SGN is used with AUTHTYPE(USERID), the exit sends the user ID to DB2 as the primary authorization ID and the RACF group ID to DB2 as the secondary ID. When the sample sign-on exit is used, there is no difference between AUTHTYPE(USERID) and AUTHTYPE(GROUP).

BUSY
this parameter is valid only with CONNECTST when setting the CICS DB2 connection NOTCONNECTED. CVDA values are:

FORCE
similar to issuing DSNC STOP FORCE; that is, any CICS transactions currently using DB2 are abnormally terminated, and the CICS DB2 attachment facility is stopped. FORCE is mutually exclusive to WAIT and NOWAIT.

NOWAIT.
makes the request asynchronous in nature. Control is returned before the request is complete. NOWAIT is mutually exclusive to WAIT and FORCE.

WAIT
The request is synchronous in nature. Control is only returned when the request is complete. WAIT is mutually exclusive to NOWAIT and FORCE.

Note that a SET DB2CONN NOTCONNECTED WAIT|NOWAIT is a quiesce stop of the CICS DB2 interface. The quiesce waits for existing transactions to finish before stopping the interface.

COMAUTHID
specifies which id should be used for security checking when using command threads. If COMAUTHid is specified, COMAUTHType may not be specified.

COMAUTHTYPE
specifies the type of id that can be used for security checking when using command threads. If COMAUTHType is specified, COMAUTHid may not be specified. CVDA values are:

CGROUP
Specifies the 8-character USERID and the connected group name as the authorization ID. The following table shows how these two values are interpreted by DB2.

IDs passed to DB2	How DB2 interprets values
CICS sign-on user ID (USERID)	Represents the primary DB2 authorization ID.
RACF-connected group name	If the RACF list of group options is not active, DB2 uses the connected group name supplied by the CICS attachment facility as the secondary DB2 authorization ID. If the RACF list of group options is active, DB2 ignores the connected group name supplied by the CICS attachment facility, but the value appears in the DB2 list of secondary DB2 authorization IDs.

To use the CGROUP option the CICS system must have SEC=YES specified in the CICS system initialization table (SIT).

If no RACF group ID is available for this USERID, an 8-character field of blanks is passed to DB2 as the group ID.

CSIGN
Specifies the SIGNID parameter of the DB2CONN should be used as the resource authorization ID.

CTERM
Specifies the terminal identification (four characters padded to eight) as an authorization ID. An authorization ID cannot be obtained in this manner if a terminal is not connected with the transaction.

If a transaction is started (using a CICS command) and has no terminal associated with it, the COMAUTHTYPE(TERM) should not be used.

CTX Specifies the transaction identification (four characters padded to eight) as the authorization ID.

COPID
The operator identification associated with the userid that is associated with the CICS transaction sign-on facility is used as the authorization ID (three characters padded to eight).

CUSERID
The 8-character userid associated with the CICS transaction is used as the authorization ID.

When the DB2 sample sign-on exit DSN3@SGN is used with AUTHTYPE(USERID), the exit sends the USERID to DB2 as the primary authorization ID and the RACF group ID to DB2 as the secondary ID. When the sample sign-on exit is used, there is no difference between COMAUTHTYPE(CUSERID) and COMAUTHTYPE(CGROUP).

COMTHREADLIM
specifies the current maximum number of command threads that the CICS DB2 attachment allows active before requests overflow to the pool.

CONNECTERROR
returns how the fact that CICS is not connected to DB2 because the adaptor is in 'standby mode' is reported back to an application that has issued a SQL request. CVDA values are:

ABEND
The application is abended with abend code AEY9.

SQLCODE
The application receives a -923 SQLCODE.

CONNECTST
sets the status of the CICS DB2 connection; that is, to start or stop the CICS DB2 connection. CVDA values are:

CONNECTED
This is equivalent to issuing DSNC STRT to start the CICS DB2 attachment. If the requested DB2 subsystem is active, control returns when CICS and DB2 have been connected. If the requested DB2 subsystem is not active, the response returned is dependant on the setting of STANDBYMODE: If DB2 is not initialized, and STANDBYMODE(NOCONNECT) is specified on the DB2CONN, INVREQ and RESP2=39 is returned. If you specify STANDBYMODE(CONNECT) or STANDBYMODE(RECONNECT), NORMAL with RESP2=38 is returned indicating that the CICS DB2 attachment is in standby mode and will connect to DB2 as soon as it becomes active.

NOTCONNECTED
NOTCONNECTED with NOWAIT means initiate quiesce stop of the connection, but return control immediately. NOTCONNECTED WAIT means that control does not return to the application until the CICS DB2 attachment has been stopped. NOTCONNECTED FORCE force stops the connection by force purging transactions currently using DB2. Control is not returned until the connection is stopped.

DB2ID
specifies the name of the DB2 subsystem that the CICS DB2 attachment should connect to. DB2ID can only be changed when CICS is not connected to a DB2 system.

MSGQUEUE1
specifies the first transient data destination to which unsolicited messages from the CICS DB2 attachment are sent.

MSGQUEUE2
specifies the second transient data destination to which unsolicited messages from the CICS DB2 attachment are sent.

MSGQUEUE3
specifies the third transient data destination to which unsolicited messages from the CICS DB2 attachment are sent.

NONTERMREL
specifies whether or not non-terminal transactions release threads for reuse at intermediate syncpoints. CVDA values are:

RELEASE
non-terminal transactions release threads for reuse at intermediate syncpoints.

NORELEASE
non-terminal transactions do not release threads for reuse at intermediate syncpoints.

PLAN
specifies the name of the plan to be used for all threads in the pool. If PLAN is specified, PLANEXITNAME may not be specified.

PLANEXITNAME
specifies the name of the dynamic plan exit to be used for pool threads. If you change the PLAN and PLANExitname while there are active transactions for the pool, the next time the transaction releases the thread, the plan/exit will be determined using the new rules. If PLANExitname is specified, PLAN may not be specified.

PRIORITY
specifies the priority of the pool thread subtasks relative to the CICS main task (QR TCB). CVDA values are:

HIGH
subtasks attain a higher priority than the CICS main task from which the subtask was generated.

EQUAL
subtasks have equal priority with the CICS main task.

LOW
subtasks have a lower priority than the CICS main task.

PURGECYCLEM
specifies in minutes the length of time of the protected thread purge cycle. The default is 0,30; that is, 30 seconds.

A protected thread is not terminated immediately when it is released. It is terminated only after two completed purge cycles, if it has not been reused in the meantime. Hence if the purge cycle is set to 30 seconds after it is released, a protected thread is purged 30 - 60 seconds after it is released. An unprotected thread is terminated when it is released (at syncpoint or end of task) if there are no other transactions waiting for a thread on that DB2ENTRY.

PURGECYCLES
specifies in seconds the length of time of the protected thread purge cycle. The default is 0,30; that is, 30 seconds.

A protected thread is not terminated immediately when it is released. It is terminated only after two completed purge cycles, if it has not been reused in the meantime. Hence if the purge cycle is set to 30 seconds after it is released, a protected thread is purged 30 - 60 seconds after it is released. An unprotected thread is terminated when it is released (at syncpoint or end of task) if there are no other transactions waiting for a thread on that DB2ENTRY.

SECURITY(REBUILD)
specifies that the CICS DB2 attachment should force all existing threads to signon again at the next thread reuse. It should be used when RACF profiles have been updated by issuing the following commands:

- CEMT PERFORM SECURITY REBUILD for RACF 1.9.2 or earlier
- TSO SETROP TS RACLIST(xxxxxxxx) REFRESH for RACF 2.1 or later

SIGNID
specifies the authorization ID to be used by the CICS DB2 attachment when signing on to DB2 for pool and DB2ENTRY threads specifying AUTHTYPE(SIGN), and command threads specifying COMAUTHTYPE(CSIGN).

STANDBYMODE
specifies the action to be taken by the CICS DB2 attachment if DB2 is not active when an attempt is made to start the connection from CICS to DB2. CVDA values are:

NOCONNECT
The CICS DB2 attachment should terminate.

CONNECT
The CICS DB2 attachment goes into 'standby mode' to wait for DB2.

RECONNECT
The CICS DB2 attachment goes into 'standby mode' and waits for DB2. Having connected to DB2, if DB2 subsequently fails the CICS DB2 attachment reverts again to standby mode and subsequently reconnects to DB2 when it comes up again.

STATSQUEUE
specifies the transient data destination for CICS DB2 attachment statistics produced when the CICS DB2 attachment is shutdown.

TCBLIMIT
specifies the maximum number of subtasks that can be identified to DB2.

THREADLIMIT
specifies the current maximum number of pool threads the CICS DB2 attachment allows active before requests are made to wait or are rejected according to the THREADWait parameter.

THREADWAIT
specifies whether or not transactions should wait for a pool thread or be abended if the number of active pool threads reach the THREADLIMIT number.

The CICS DB2 attachment issues a unique abend code AD3T, and message DFHDB2011 when THREADWAIT=NO is coded and the number of pool threads is exceeded. CVDA values are:

TWAIT
If all threads are busy, a transaction must wait until one becomes available. A transaction can wait as long as CICS allows it to wait, generally until a thread becomes available.

NOTWAIT
If all threads are busy the transaction is terminated with an abend code AD3T.

Notes:

1. When you change the value of AUTHID, AUTHTYPE, COMAUTHID, COMAUTHTYPE or SIGNID, a surrogate user security check is invoked if security is active. This ensures that the userid under which SET is being executed is authorized to act on behalf of the userid being set.
2. When the SET DB2CONN command is specified all parameters except the DB2ID (the connected subsystem) can be set when the CICS DB2 attachment is active. *DB2ID can only be changed by stopping and restarting the attachment*.
3. If you change the PLAN and PLANEXITNAME while there are active transactions for that entry, or the pool, the next time the transaction releases the thread, the plan/exit will be determined using the new rules.

Conditions

NORMAL
RESP2 values:

38 Waiting for DB2
(this may occur following a CONNECTST with a CVDA of CONNECT)

NOTAUTH
The user associated with the issuing task is not authorized to use this command, or is not authorized to access this resource.

RESP2 values:

100 Command authorization failure
102 Surrogate authorization failure
103 Authtype authorization failure

NOTFND
RESP2 values:

1 There is no DB2CONN currently installed .

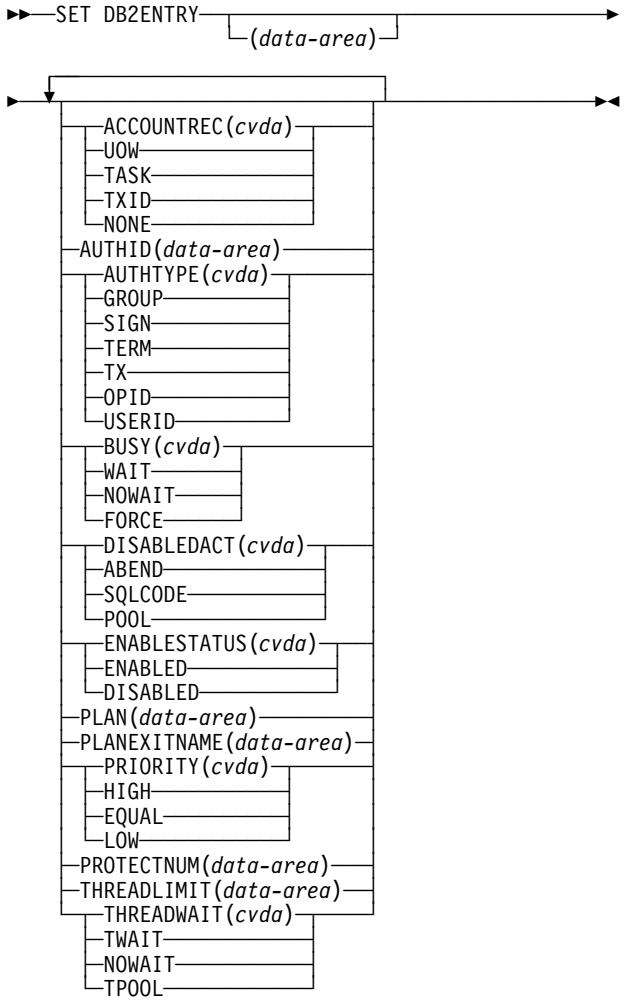
INVREQ
RESP2 values:

1 Invalid ACCOUNTREC value
2 Invalid AUTHTYPE value
3 Invalid BUSY value
4 Invalid COMAUTHTYPE value
5 Invalid CONNECTERROR value
6 Invalid CONNECTST value
7 Invalid NONTERMREL value
9 Invalid PRIORITY value
10 Invalid SECURITY value
11 Invalid STANDBYMODE value
12 Invalid THREADWAIT value
13 Bad characters in AUTHID
14 Bad characters in COMAUTHID
15 Bad characters in DB2ID
16 Bad characters in MSGQUEUE1
17 Bad characters in MSGQUEUE2
18 Bad characters in MSGQUEUE3
19 Bad characters in PLAN
20 Bad characters in PLANEXITNAME
21 Bad characters in SIGNID
22 Bad characters in STATSQUEUE
23 Both AUTHID and AUTHTYPE specified
24 Both COMAUTHID and COMAUTHTYPE specified
25 STANDBYMODE(NOCONNECT) and CONNECTERROR(SQLCODE) specified or CONNECTERROR(SQLCODE) specified when STANDBYMODE is NOCONNECT.
26 Both PLAN and PLANEXITNAME specified.
27 Invalid ACCOUNTREC value
28 COMTHREADLIM exceeds TCBLIMIT or COMTHREADLIM > 2000 or COMTHREADLIM < 0
29 Purge cycle too low; that is, < 30 seconds
- or Purge cycle minutes < 0
- or Purge cycle seconds < 0
- or Purge cycle minutes > 59
- or Purge cycle seconds > 59

32 Tcblimit > 2000 or Tcblimit < 4
33 Threadlimit exceeds tcblimit or Threadlimit > 2000 or Threadlimit < 3
34 Already connected
39 DB2 not active
40 Insufficient authorization
41 Connection error
42 Invalid init parms
43 DB2ID cannot be set, conn active
44 DB2CONN partially discarded
46 SET NOTCONNECTED when the FORCE or WAIT option has been specified, but this transaction is itself using the CICS DB2 interface.

SET DB2ENTRY

Sets the attributes of a particular DB2ENTRY used to define resources to be used by a specific transaction or by a group of transactions when accessing DB2.



Conditions: NOTAUTH, NOTFND, INVREQ

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

Description

The entry is identified by the name it was defined with in CEDA. For RCTs migrated to the CSD, the name is the name of the first transaction on the DSNCRCT TYPE=ENTRY statement.

Options

ACCOUNTREC

Specifies the minimum amount of DB2 accounting required for transactions using pool threads. The specified minimum may be exceeded as described in the following options. CVDA values are:

NONE

No accounting records are required for transactions using pool threads.

DB2 produces at least one accounting record for each thread when the thread is terminated. Authorization changes additionally cause accounting records to be produced.

TXID

The CICS DB2 attachment facility causes an accounting record to be produced when the transid using the thread changes.

Because pool threads are typically used by a number of different transaction IDs, there is an increased chance that a transaction containing multiple units of work (UOWs) will use a different thread for each UOW (assuming the thread is released at syncpoint). In this case an accounting record may be produced per UOW.

TASK

The CICS DB2 attachment facility causes a minimum of one accounting record for each CICS task to be produced.

A transaction containing multiple UOWs (assuming the thread is released at syncpoint) may use a different thread for each of its UOWs. The result may be an accounting record produced for each UOW.

UOW

The CICS DB2 attachment facility causes an accounting record to be produced for each UOW, assuming that the thread is released at the end of the UOW.

AUTHID

specifies the id to be used for security checking when using this DB2ENTRY. If AUTHId is specified, AUTHTYpe may not be specified.

AUTHTYPE

returns the type of id that can be used for security checking when using this DB2ENTRY. If AUTHTYpe is specified, AUTHid may not be specified. CVDA values are:

GROUP

Specifies the 8-character USERID and the connected group name as the authorization ID. The following table shows how these two values are interpreted by DB2.

	IDs passed to DB2	How DB2 interprets values
	CICS sign-on user ID (USERID)	Represents the primary DB2 authorization ID.
	RACF-connected group name	If the RACF list of group options is not active, DB2 uses the connected group name supplied by the CICS attachment facility as the secondary DB2 authorization ID. If the RACF list of group options is active, DB2 ignores the connected group name supplied by the CICS attachment facility, but the value appears in the DB2 list of secondary DB2 authorization IDs.

To use the GROUP option the CICS system must have RACF external security SEC=YES specified in the CICS system initialization table (SIT).

If no RACF group ID is available for this USERID, an 8-character field of blanks is passed to DB2 as the group ID.

SIGN
: Specifies the SIGNID parameter of the DB2CONN should be used as the resource authorization ID.

TERM
: Specifies the terminal identification (four characters padded to eight) as an authorization ID. An authorization ID cannot be obtained in this manner if a terminal is not connected with the transaction.

  If a transaction is started (using a CICS command) and has no terminal associated with it, AUTHTYPE(TERM) should not be used.

TX
: Specifies the transaction identification (four characters padded to eight) as the authorization ID.

OPID
: The operator identification associated with the CICS transaction is used as the authorization ID (three characters padded to eight).

USERID
: The 8-character USERID associated with the CICS transaction is used as the authorization ID.

  When the DB2 sample sign-on exit DSN3@SGN is used with AUTHTYPE(USERID), the exit sends the user ID to DB2 as the primary authorization ID and the RACF group ID to DB2 as the secondary ID. When the sample sign-on exit is used, there is no difference between AUTHTYPE(USERID) and AUTHTYPE(GROUP).

BUSY(Cvda)
specifies what CICS is to do if a SET DB2ENTRY DISABLED is issued and the entry is busy when the set command is issued. CVDA values are:

WAIT
: CICS must wait for all activity on the DB2ENTRY to be quiesced before setting the DB2ENTRY disabled. CICS then returns control to the application.

  Note that when a DB2ENTRY is quiescing, all existing transactions are allowed to complete. Transactions already queued against the entry are also allowed to complete. New transactions that try to access the DB2ENTRY are routed to the POOL, or abended, or sent a SQLCODE depending on the setting of DISABLEDACT.

NOWAIT
: is the same as WAIT except that control returns to the application as soon as the SET DISABLED request is queued.

FORCE
: all tasks using the DB2ENTRY, and those queued against the DB2ENTRY are forcepurged. The DB2ENTRY is then DISABLED and control returns to the application.

DISABLEDACT
specifies what CICS is to do with new transactions that access a DB2ENTRY when it has been disabled or disabling. CVDA values are:

POOL
: The CICS DB2 attachment facility routes the request to the pool. Message DFHDB2072 is sent to the transient data destination specified by MSGQUEUEn on the DB2CONN for each transaction routed to the pool.

ABEND
: The CICS DB2 attachment facility abends the transaction.The abend code is AD26.

SQLCODE
: An SQLCODE is returned to the application indicating that the DB2ENTRY is disabled.

ENABLESTATUS(cvda)
specifies whether the DB2ENTRY can be accessed by applications. CVDA values are:

ENABLED
: The DB2ENTRY can be accessed by applications.

DISABLED
: The DB2ENTRY cannot be accessed by applications. A DB2ENTRY has to be disabled before it can be reinstalled or discarded.

PLAN
specifies the name of the plan to be used for this DB2ENTRY.

If PLAN is specified, PLANEXITNAME cannot be specified.

PLANEXITNAME
specifies the name of the dynamic plan exit to be used for this DB2ENTRY. If you change the PLAN and PLANExitname while there are active transactions for the DB2ENTRY the next time the transaction releases the thread, the plan/exit will be determined using the new rules. If PLANExitname is specified, PLAN cannot be specified.

PRIORITY
specifies the priority of the thread tasks for the DB2ENTRY relative to the CICS main task (QR TCB). CVDA values are:

HIGH
subtasks attain a higher priority than the CICS main task from which the subtask was generated.

EQUAL
subtasks have equal priority with the CICS main subtask.

LOW
subtasks have a lower priority than the CICS main task.

PROTECTNUM
specifies the maximum number of protected threads for this DB2ENTRY.

THREADLIMIT
specifies the maximum number of threads for this DB2ENTRY that the CICS DB2 attachment allows active before requests are made to wait or are rejected.

THREADWAIT
specifies whether or not transactions should wait for a DB2ENTRY thread, be abended, or overflow to the pool should the number of active DB2ENTRY threads reach the THREADLimit number. CVDA values are:

TWAIT
If all threads are busy, a transaction waits until one becomes available.

NOTWAIT
If any threads are busy, a transaction is terminated with an abend code AD2P.

TPOOL
If all threads are busy, the transaction is diverted to use the pool of threads. If the pool is also busy, and NOTWAIT has been specified for the THREADWAIT parameter on the DB2CONN. The transaction is terminated with abend code AD3T.

Notes:

1. When you change the value of AUTHId or AUTHType, a surrogate user security check is invoked if security is active. This ensures that the userid under which SET is being executed is authorized to act on behalf of the userid being set.
2. All parameters on SET DB2ENTRY can be set whilst the CICS DB2 attachment is active and the transactions are active.

Conditions

NOTAUTH
The user associated with the issuing task is not authorized to use this command, or is not authorized to access this resource.

RESP2 values:

100 Command authorization failure
101 Resource authorization failure
102 Surrogate authorization failure
103 Authtype authorization failure

NOTFND
RESP2 values:

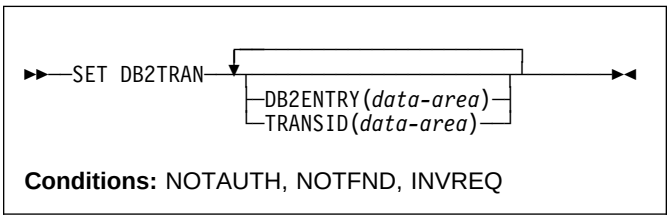
1 There is no DB2ENTRY currently installed with the specified name.

INVREQ
RESP2 values:

2 Invalid action value
3 Invalid Authtype value
4 Invalid busy value
5 Invalid enablestatus value
7 Invalid priority value
8 Invalid Threadwait value
9 Bad characters in Authid
10 Bad characters in Plan
11 Bad characters in Planexitname
12 Both Authid and Authtype specified
13 Both Plan and Planexitname specified
14 Entry is disabling
15 Protectnum greater than Threadlimit or protectnum < 0 or protectnum > 2000
16 Threadwait must be tpool with Threadlimit=0
17 Threadlimit > 2000 or Threadlimit < 0 or Threadlimit > TCBLIMIT
18 Invalid Accountrec value
19 SET DISABLED when the FORCE or WAIT option has been specified, but this transaction is itself using the DB2ENTRY.

SET DB2TRAN

Sets the attributes of a particular DB2TRAN associated with a DB2ENTRY.



Conditions: NOTAUTH, NOTFND, INVREQ

Description

The DB2TRAN is identified by the name it was defined with in CEDA. For RCTs migrated to the CSD, the name matches the name of the transaction for which the DB2TRAN is being created.

If a transid is specified on a DB2ENTRY when the DB2ENTRY is installed, CICS installs a DB2TRAN named DFHxxxx, where xxxx is the transid.

Options

DB2ENTRY
specifies the name of the DB2Entry to which this DB2Tran refers; that is, the DB2ENTRY with which this additional transid should be associated.

TRANSID
specifies the transaction id to be associated with the entry. The transaction id can include wildcard characters (see the *CICS Resource Definition Guide* for information about use of wildcard characters). If you change TRANSID for a DB2TRAN while the attachment is active, all transactions with a thread continue to use the thread from that entry until it is released for reuse. When that transaction issues the next SQL statement, the thread is acquired from the entry or pool based upon the new definition.

DB2TRAN parameters may be set at any time.

Conditions

NOTAUTH
The user associated with the issuing task is not authorized to use this command, or is not authorized to access this resource.

RESP2 values:

100 Command authorization failure
101 Resource authorization failure
102 Surrogate authorization failure
103 Authtype authorization failure

NOTFND
RESP2 values:

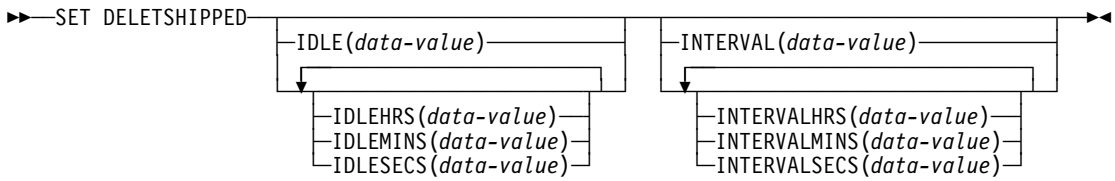
1 There is no DB2TRAN currently installed with the specified name.

INVREQ
RESP2 values:

2 Bad characters in TRANSID name
3 Transid already exists in another installed DB2TRAN
4 Bad characters in DB2ENTRY name

SET DELETSHIPPED

Change the system settings that control automatic deletion of shipped terminal definitions.



Conditions: INVREQ, NOTAUTH

Description

The SET DELETSHIPPED 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 that requires the terminal is waiting to be attached. For more information about shipped definitions, see the *CICS Intercommunication Guide* and *CICS 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, INTERVALMINS, 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 DELETSHIPPED command, described on page "PERFORM DELETSHIPPED" on page 231.

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:

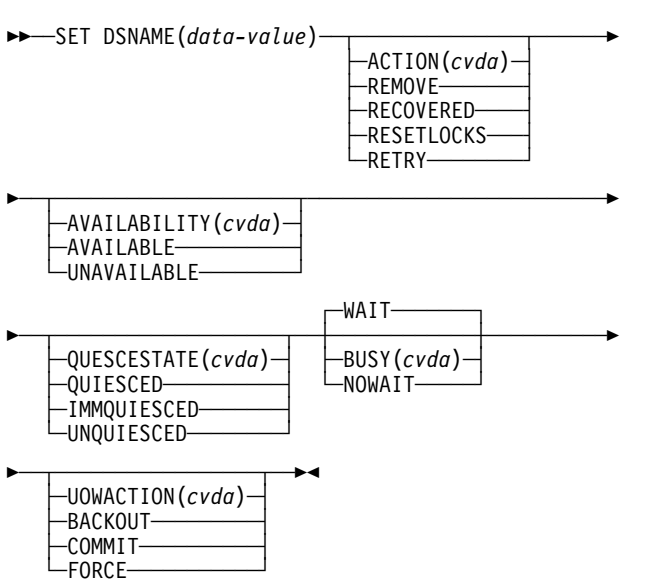
1 The INTERVAL value is invalid.
2 The INTERVALHRS value is not in the range 0-99.
3 The INTERVALMINS value is invalid.
4 The INTERVALSECS value is invalid.
5 The IDLE value is invalid.
6 The IDLEHRS value is not in the range 0-99.
7 The IDLEMINS value is invalid.
8 The IDLESECS value is invalid.

NOTAUTH
RESP2 values:

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

SET DSNAME

Change information relating to an external data set, including actions that apply to all UOWs that access this dataset.



Conditions: DSNNOTFOUND, INVREQ, IOERR, NOTAUTH, SUPPRESSED

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

Description

With the SET DSNAME command, you can:

- Tell CICS that a data set is no longer required on the local system.
- Set the “backup while open” (BWO) attributes of the data set to the ‘forward recovered’ state by updating the ICF catalog. This indicates that a forward recovery has taken place.
- Mark a VSAM data set as quiesced, or unquiesced, throughout the sysplex.
- Make a VSAM data set available or unavailable to a CICS region. (The availability function does not operate across the sysplex—a SET DSNAME (...) AVAILABILITY(...) command is effective only within the CICS region in which it is issued.)
- Retry all UOW log records that are shunted due to the failures of this data set (other than in-doubt failures).
- Force any UOWs that are shunted due to in-doubt failures, and which have updated this data set, to complete.

- Purge shunted UOW log records that hold retained locks (other than those due to in-doubt failures) for any records in the data set, and release the retained locks,
- Cancel any attempt to recover lost RLS locks for the data set, using the UOWACTION and ACTION(RESETLOCKS) options.

For information about shunted UOW log records, see the *CICS Recovery and Restart Guide*.

The options and CVDAs for the SET DSNAME command are subject to the following rules relating to the order of processing and the combinations of keywords and multiple keywords on the same command:

- If REMOVE is specified, no other attribute is allowed.
- Options are processed in the following order:
  1. RECOVERED
  2. UNQUIESCED
  3. AVAILABLE
  4. RETRY
  5. UOWACTION
  6. RESETLOCKS
  7. UNAVAILABLE
  8. QUIESCED.

If you specify RETRY, you should not also specify UNAVAILABLE or QUIESCED, because this could cause backout retries to fail.

If you combine UNQUIESCED with any other attributes, you should also specify BUSY(WAIT), so that later options do not cause the command to fail because the data set is not unquiesced.

Some of the options of a data set cannot be specified until the first file that references the data set has been opened. Where an attribute is not valid until a file has been opened, the INVREQ condition is returned. Note that QUIESCESTATE is an attribute that can be used before any files have been opened against the specified data set.

Options

ACTION(*cvda***)**
specifies the action to be taken on the data set. CVDA values are:

RECOVERED
This data set has been restored from a backup version and forward recovery has been run and completed successfully. CICS attempts to update the BWO attributes of the data set in the ICF catalog using DFSMS™ callable services. The command is used by the database administrator to update the BWO attributes in the ICF catalog if the forward recovery log apply utility does not do so, or if the database administrator finds that there has been no update since the backup copy was made. This would mean that no forward recovery is needed. If the BWO attributes of the data set are not updated after restoring a backup copy, a subsequent file open fails because the data set is still marked as down-level in the ICF catalog.

For information about DFSMS callable services see *OS/390 Security Server (RACF) Security Administrator's Guide*.

REMOVE
A data set is no longer required on the local system. Before you can issue a SET DSNAME REMOVE command, the data set must have a FILECOUNT of zero. If you specify REMOVE, you must not specify any other option.

Removing temporary data sets: If you have an application that creates temporary data sets, it is most important that you remove the associated data set name blocks when the data sets are no 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 or initial 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” on page 262 for an illustration of how you can identify and remove unwanted data set name blocks.

RESETLOCKS (VSAM only)
Purges shunted UOW log records for backout-failed and commit-failed UOWs that hold locks on this data set, and releases the retained locks.

- Backout-failed UOWs are those that failed during backout processing.
- Commit-failed UOWs are those that have updated RLS data sets, and have failed to release locks during the second phase of 2-phase commit syncpoint processing.

If you specify this option, you are accepting backout failure and some loss of data integrity rather than retaining locks and delaying transactions, and therefore it should be used only as a last resort.

For backout-failed and commit-failed UOWs that hold locks on the data set, all records relating to this data set are removed from the system log and all retained record locks held by this CICS for the data set are released. Diagnostic messages are written to the CSFL transient data queue for each backout-failed log record that is removed as a result of the RESETLOCKS operation.

You might choose to use RESETLOCKS if backout-failed or commit-failed log records are holding up lost locks recovery for the data set, and there is no other way of resolving them.

Notes:

1. This option does not apply to shunted *in-doubt* UOWs. You should try to resolve the shunted in-doubt UOWs that hold locks on the data set in other ways before issuing RESETLOCKS; for example, by using COMMIT, BACKOUT, or FORCE (see the UOWACTION option).
2. RESETLOCKS can fail during the commit phase (for example, if an error occurs while CICS is trying to release the RLS locks), in which case the UOWs revert to being shunted as commit-failed UOWs.

RETRY
: Shunted UOW log records, caused by failed backout and commit processing for this data set, should be retried. This is similar in concept to the SET CONNECTION RESYNC command, but applies to backout-failed and commit-failed UOWs only, and not to in-doubt UOWs.

  You should use RETRY when the data set has shunted backout-failed or commit-failed UOWs associated with it, and you believe that some or all of the data set problems are either transient or have been resolved. If the data set was damaged in some way, it must have been repaired (recreated) and made available for RETRY to work successfully.

  Messages issued at the time of a data set failure, and which cause UOWs to be shunted, recommend the actions required to recover from the failure.

  RETRY does not harm data integrity, and can be used safely at any time to enable some failed recovery work to complete.

AVAILABILITY(*cvda***) (VSAM only)**
: specifies whether the data set is to be marked, in this CICS region, as available or unavailable for use. This command sets or unsets the availability indicator, which is a local flag that a CICS region maintains in a data set name block (DSNB) for each data set. CVDA values are:

  AVAILABLE
  : The data set is available. CICS can issue both RLS and non-RLS open requests for this data set.

  UNAVAILABLE
  : The data set is unavailable. The data set cannot be opened in either RLS or non-RLS modes.

BUSY(*cvda***) (RLS only)**
: specifies whether CICS should wait when requested to quiesce or unquiesce the data set, provided QUIESCESTATE has also been specified. It is ignored if QUIESCESTATE is not specified. CVDA values are:

  NOWAIT
  : CICS returns control to the application immediately, having started the quiesce or unquiesce operation asynchronously. You can use INQUIRE DSNAME QUIESCESTATE to check whether the quiesce or unquiesce has completed.

  WAIT
  : CICS returns control to the application only when the data set has been quiesced or unquiesced throughout the sysplex, or when it has failed to do so. If a quiesce is not completed within the time specified in the QUIESTIM system initialization parameter, the quiesce times out. See the QUIESTIM system initialization parameter in the *CICS System Definition Guide*. If you specify WAIT, or allow it to default, you should ensure that your program handles an AEXY abend in case the DTIMOUT value is not high enough to allow your task to wait for completion.

DSNAME(*data-value***)**
: specifies the name of the data set. It can be up to 44 characters long, and is defined to CICS in the DSNAME operand of the CEDA DEFINE FILE command.

QUIESCESTATE(*cvda***) (RLS only)**
: specifies the RLS quiesce state of the data set. The state is set in the ICF catalog entry for the data set when the operation has completed. CVDA values are:

  IMMQUIESCED
  : All existing CICS files open in RLS mode throughout the sysplex are closed and the data set is marked as quiesced in the ICF catalog. Each CICS in the sysplex abends all in-flight UOWs that are accessing the data set before closing files, causing in-flight UOWs to back out. Any UOWs that fail backout are shunted. No files can open in RLS mode against this data set, but non-RLS open requests are permitted (although opens for update are not possible in non-RLS mode if the data set has retained RLS locks).

    In addition to closing open files, IMMQUIESCED sets the file state to UNENABLED if it was ENABLED. A subsequent SET DSNAME UNQUIESCED command restores the file state to ENABLED, provided it was set UNENABLED by a QUIESCED or IMMQUIESCED action, but *not* if the UNENABLE state is because of some other event. This state change is recorded in the CICS global catalog.

  Note: Using the IMMQUIESCED option causes any tasks currently using the data set to be terminated immediately, using the CICS task FORCEPURGE mechanism. In some extreme cases, CICS may terminate abnormally. For this reason, setting a data set as quiesced using the IMMQUIESCED option should be restricted to exceptional circumstances.

QUIESCED
: All existing CICS files open in RLS mode throughout the sysplex are closed and the data set is marked as quiesced in the ICF catalog. Each CICS in the sysplex waits until all in-flight UOWs that are accessing the data set have reached syncpoint before closing the files; that is, the UOWs are either:

  - Successfully committed, or
  - Successfully backed out, or
  - Shunted because of an in-doubt failure, or
  - Shunted because of a failed backout, or
  - Shunted because of a failed commit.

  Note: If you specify QUIESCED with WAIT (the default), all tasks in all CICS regions in the sysplex must have reached syncpoint before the files are closed, allowing your command to complete. You must ensure that the DTIMOUT value for the transaction issuing the QUIESCED command is sufficient to allow for this, otherwise the transaction abends with an AEXY abend. The QUIESCE operation is allowed to run until completed or until the timeout value set by the QUIESTIM system initialization parameter, (for which the default is 4 minutes), is reached.

  No files can open in RLS mode against this data set, but non-RLS open requests are permitted (although opens for update are not possible in non-RLS mode if the data set has retained RLS locks).

  In addition to closing open files, QUIESCED sets the file state to UNENABLED if it was ENABLED. A subsequent SET DSNAME UNQUIESCED command restores the file state to ENABLED, provided it was set UNENABLED by a QUIESCED or IMMQUIESCED action, but *not* if the UNENABLE state is because of some other event. This state change is recorded in the CICS global catalog.

UNQUIESCED
: The data set is marked as unquiesced in the ICF catalog. RLS or non-RLS opens can be issued against this data set, the access mode (RLS or non-RLS) being established by the first open. After the first successful open request, subsequent open requests in the same mode as the first open only are permitted.

  If a file has been set UNENABLED by an earlier SET DSNAME IMMQUIESCED or QUIESCED command, UNQUIESCED restores the file state to ENABLED. This state change is recorded in the CICS global catalog.

UOWACTION(*cvda***)**
: specifies the action to be taken for shunted in-doubt UOWs. CVDA values are:

BACKOUT
: All shunted in-doubt UOWs that hold locks on this data set should be backed out.

COMMIT
: All shunted in-doubt UOWs that hold locks on this data set should be committed.

FORCE
: All shunted in-doubt UOWs that hold locks on this data set should be FORCED to back out or commit, as specified by the ACTION attribute defined on the transaction resource definition.

Conditions

DSNNOTFOUND
RESP2 values:

1 The named data set cannot be found.
15 RECOVERED was specified, but the data set was not found.

INVREQ
RESP2 values:

3 ACTION has an invalid CVDA value.
10 REMOVE was specified, but the data set is associated with a file definition.
12 REMOVE was specified with another option. If you specify REMOVE, it must be the only option present on the command.
13 REMOVE was specified but a lock was held on the data set by another INQUIRE or SET DSNAME command, or by CICS file control processing.
14 RECOVERED was specified but CICS is not configured to support “backup while open” (BWO). Check that you have a version of MVS/DFP™, DFHSM, and DFDSS that supports BWO.
16 RECOVERED was specified but the data set has not been opened during this CICS session, so the BWO attributes in the ICF catalog cannot be set.
17 RECOVERED was specified for a BDAM data set, or a VSAM path. This is not supported.
18 RECOVERED was specified for a VSAM base data set that has FCTs open. This is not allowed.
19 RECOVERED was specified for an unknown data set, or the data set was not in the ‘forward recovered’ state.
29 QUIESCESTATE is specified, but the operation is not supported because RLS=NO is specified as a system initialization parameter, or because DFSMS 1.3 or later is not installed.
30 QUIESCESTATE has an invalid CVDA value.
31 BUSY has an invalid CVDA value.
33 AVAILABILITY has an invalid CVDA value.
34 A QUIESCESTATE value of QUIESCED or IMMQUIESCED is specified, but is rejected by SMSVSAM either because a quiesce or unquiesce

is already taking place, or because DFSMSdss is currently taking a backup copy of the data set.

36 A QUIESCESTATE value of UNQUIESCED is specified, but is rejected by SMSVSAM either because an unquiesce is already taking place, or because DFSMSdss™ is currently taking a backup copy of the data set.

39 AVAILABILITY, QUIESCESTATE, RESETLOCKS, or RETRY is specified for a data set that is a BDAM data set.

40 The CICS control block (DSNB) describing the data set has been deleted (by the REMOVE option) by another task before CICS could process this SET command.

41 QUIESCESTATE is specified for a data set that is not known to DFSMS as a VSAM data set.

42 An invalid CVDA is specified for UOWACTION.

43 A QUIESCESTATE value of QUIESCED or IMMQUIESCED is specified without NOWAIT, and the issuing task has updated the data set, or is browsing the data set, in the same unit of work. This is not allowed because:

- For QUIESCED, this would result in a deadlock.
- For IMMQUIESCED, this would result in the issuing task being purged.

44 A SET DSNAME REMOVE command has been issued by another task. This has been detected after this SET DSNAME command was issued, but before the AVAILABILITY option is processed.

46 BKOUTSTATUS is specified with a value other than NORMALBKOUT (BKOUTSTATUS is obsolete).

47 No file has been opened against the data set since the last cold start of this CICS region, or since the first file definition was installed for the data set.

IOERR

RESP2 values:

20 RECOVERED was specified but an error was raised on accessing the ICF catalog. Ensure that the specified data set is on an SMS-managed DASD and is known to the SMS subsystem.

21 RECOVERED was specified but an error was raised by the CICS table manager program.

35 QUIESCESTATE is specified but the SMSVSAM server is not available.

40 QUIESCESTATE is specified, and an unexpected error occurred in DFSMS.

48 The specified operation cannot be completed because the data set is migrated. Recall the data set and reissue the command.

49 An error was raised by DFSMS when reading the ICF Catalog to establish the base data set name.

NOTAUTH

RESP2 values:

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

SUPPRESSED

RESP2 values:

37 A QUIESCESTATE value of QUIESCED or IMMQUIESCED is specified, but the quiesce of the data set is cancelled by another participating CICS region. This could be for one of the following reasons:

- A user issued a SET DSNAME UNQUIESCED command.
- An XFCVSDS global user exit program suppressed the quiesce.
- An XFCSREQ global user exit program suppressed the close of a file that is open against the data set.

38 A QUIESCESTATE value of QUIESCED or IMMQUIESCED is specified, but the quiesce of the data set is cancelled by this CICS region because the quiesce operation timed out. This is probably because of a long-running transaction on another participating CICS region preventing the close of a file that is open against the data set.

Another reason for the timeout could be that one or more regions are very busy. If this occurs too frequently, you can modify the timeout period (from the default of 240 seconds) by specifying a longer period using the QUIESTIM system initialization parameter.

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 reusing 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 or initial 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.

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

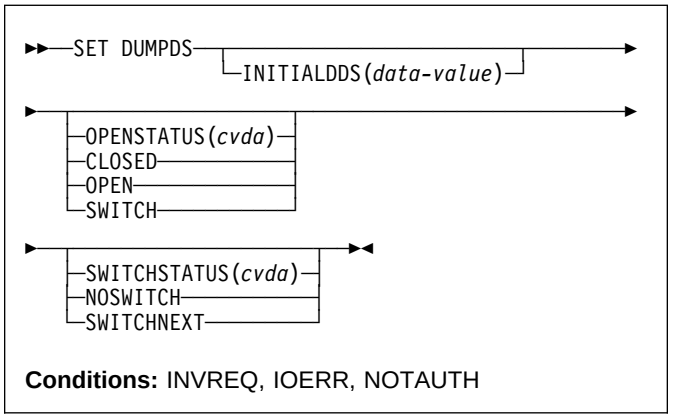
```
*PROCESS XOPTS(SP);
 REMOVE:PROC OPTIONS(MAIN);
 DCL PLIXOPT STATIC EXTERNAL CHAR(10) VAR INIT('NOSTAE');
 DCL TEMPDSN CHAR(44);
/*                                                    */
/*         Find name of temporary data set if not known  */
         EXEC CICS INQUIRE FILE('TEMPFILE') DSNAME(TEMPDSN);
*/         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.



Conditions: INVREQ, IOERR, NOTAUTH

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: If a CICS system is initialized without any transaction dump data sets, only the last two functions will be 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 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 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.

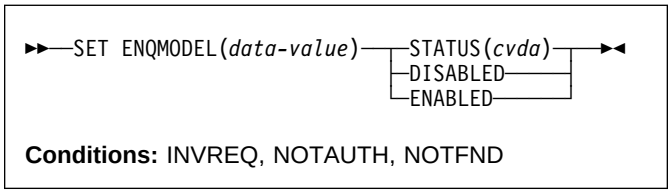
Examples

```
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 ENQMODEL

Change the status of an ENQMODEL definition.



Description

The SET ENQMODEL command allows you to ENABLE or DISABLE ENQMODEL resources installed on the local system. An ENQMODEL must be enabled to allow matching EXEC ENQ requests to be processed. It must be disabled to allow a more specific ENQMODEL to be enabled.

ENQMODELs forming nested generic enqnames must be enabled in order, from the most to the least specific. For example, enable ABCD* then ABC* then AB*.
If you attempt to enable a more specific ENQMODEL when a less specific enqmodel is already enabled, the result is that msg NQ0107 is issued and INVREQ is returned to the caller.

In this case you may need to disable one or more less specific ENQMODELs to allow a more specific ENQMODEL to be enabled. You will then be able to enable the less specific ENQMODELs again.

You cannot enable/disable an ENQMODEL which is in the waiting state. If attempted, INVREQ is returned to the caller.

Options

ENQMODEL(*data-value***)**
specifies the 8-character identifier of the resource definition.

STATUS(*cvda***)**
specifies the action to be taken on the ENQMODEL. CVDA values are:

ENABLED
If the ENQMODEL is DISABLED, it will be ENABLED. Once enabled, matching ENQ requests will be processed in the normal way.

DISABLED
The ENQMODEL will be put into the WAITING state until there are no enqueues in the local system which match the ENQNAME pattern. It will then be DISABLED. Once disabled, matching ENQ requests will be rejected, and the issuing task will be abended.

Conditions

INVREQ
RESP2 values:

2 The attempt to enable/disable an ENQMODEL failed, because a more generic ENQMODEL is enabled.
3 STATE has an invalid CVDA value.
4 The ENQMODEL is in the WAITING state

NOTAUTH
RESP2 values:

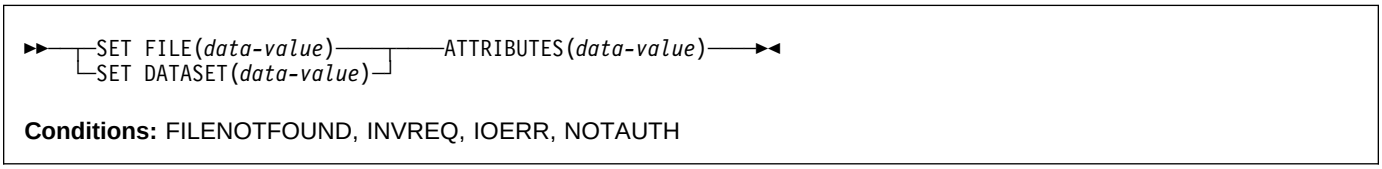
100 The user of the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The specified ENQMODEL is not installed on this system.

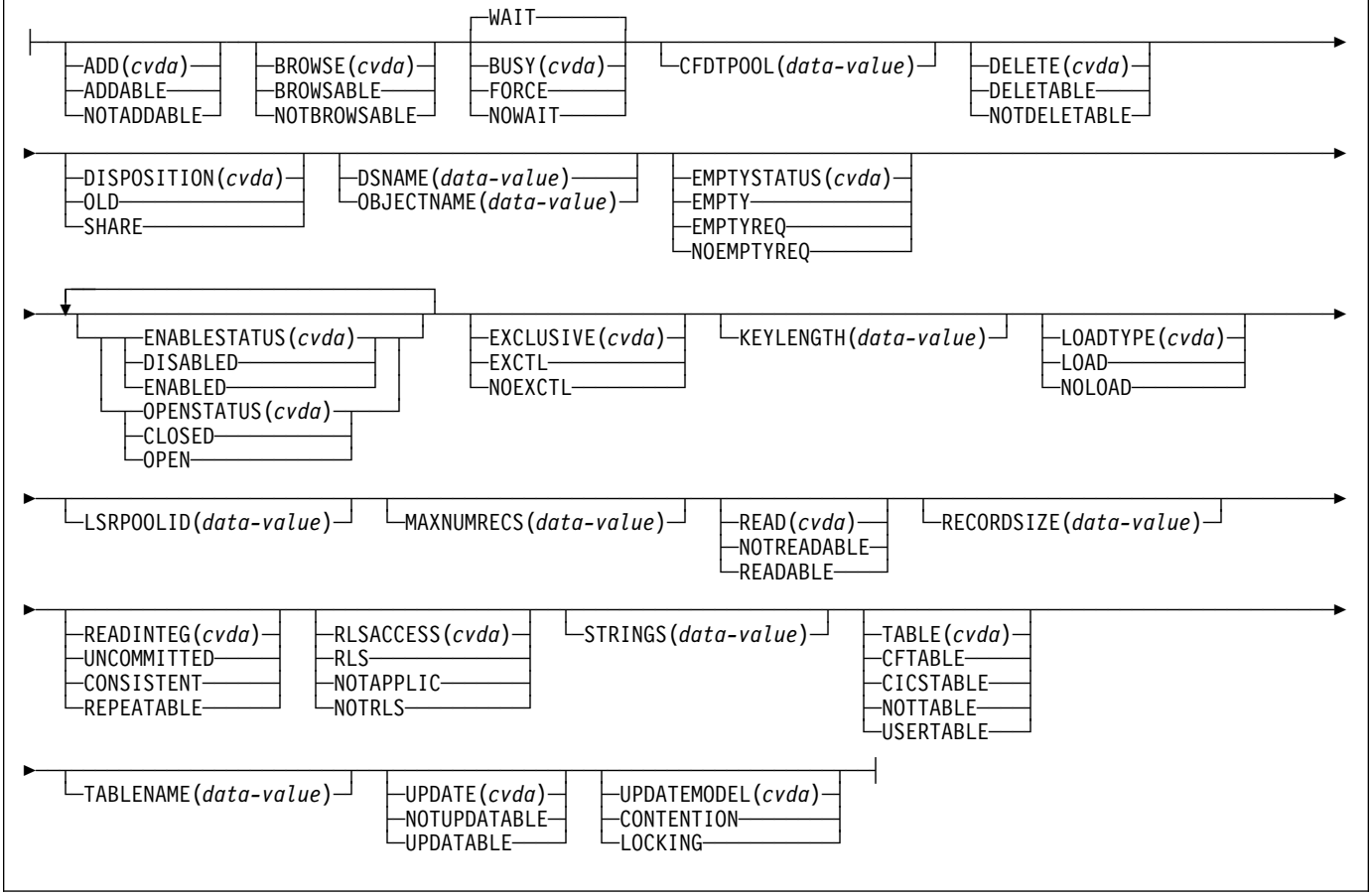
SET FILE

Change attributes of a VSAM or BDAM file, including files that refer to CICS shared data tables and coupling facility data tables.



Conditions: FILENOTFOUND, INVREQ, IOERR, NOTAUTH

FILE attributes:



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

Note: This command replaces the SET DATASET command. The DATASET keyword is supported by the translator as a synonym for FILE, but you should use FILE for all new applications. Similarly, OBJECTNAME is supported as a synonym for DSNAME.

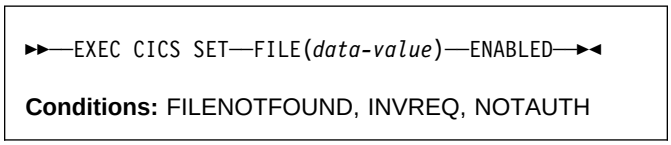
Any combination of the options can be set on one command. All changes, other than to close and disable the file, require that the file be in a CLOSED state, with an ENABLESTATUS of either DISABLED or UNENABLED, and they do not take effect until the file is next opened.

| You can use the SET FILE command to set combinations of attributes that are relevant to more than one file type, to simplify switching between different types of file. Attributes that are not relevant to the current type of file are ignored. You can use this capability to set up dual-purpose file definitions, for example, by defining both local and remote attributes, or set attributes that make it easy to switch the file from accessing a user-maintained data table within a single MVS image to accessing a coupling facility data table within a Parallel Sysplex.

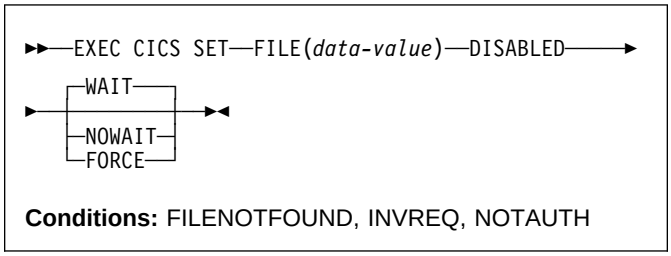
| If a coupling facility data table already exists, and the table
| attributes specified on the SET FILE command do not match
| those with which it was created, an attempt to open the file
| fails with an error message.

| If you use the SET FILE command to switch the file from
| referencing a coupling facility data table to a different object
| (for example from CFTABLE to NOTTABLE), the CFDT is
| not deleted and remains in existence in its pool (the coupling
| facility list structure).

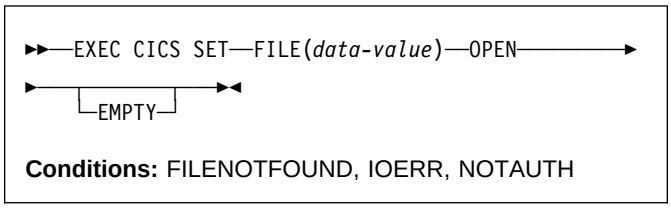
The requested changes are applied in the following order: NOEMPTYREQ, CLOSED, DISABLED, miscellaneous, OPEN, ENABLED.



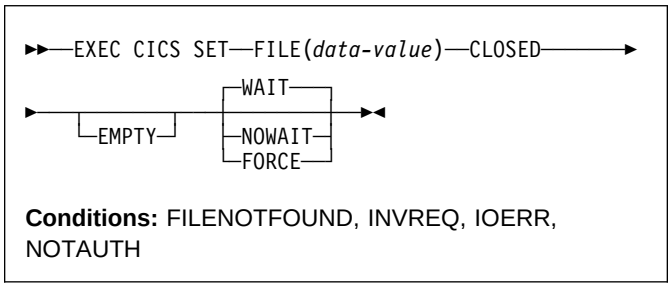
Conditions: FILENOTFOUND, INVREQ, NOTAUTH



Conditions: FILENOTFOUND, INVREQ, NOTAUTH



Conditions: FILENOTFOUND, IOERR, NOTAUTH



Conditions: FILENOTFOUND, INVREQ, IOERR, NOTAUTH

Description

The SET FILE command allows you to change some of the attributes of a named VSAM or BDAM file. A security check is made and an unauthorized command attempt is given a NOTAUTH response. If any retained locks are associated with the file, the only attributes that you are allowed to change are the ENABLESTATUS and the OPENSTATUS. Attempting to specify any other attribute when there are retained locks causes an INVREQ condition to be raised.

Options

ADD(*cvda***)**
specifies whether new records are to be added to the file. CVDA values are:

ADDABLE
New records are to be added to the file.

NOTADDABLE
New records are not to be added to the file.

BROWSE(*cvda***)**
specifies whether the file is to be browsable. CVDA values are:

BROWSABLE
The file is to be browsable.

NOTBROWSABLE
The file is not to be browsable.

BUSY(*cvda***)**
specifies what CICS is to do if the file is in use when you issue the SET command. The BUSY option is valid only for requests to SET the file DISABLED or CLOSED, and is ignored for any other request. CVDA values are:

FORCE
All tasks using the file are abended, the file is immediately DISABLED or CLOSED, and control returns to the issuing application.

NOWAIT
The same as WAIT, except that CICS returns control to the issuing application as soon as the SET request has been queued.

WAIT
CICS is to wait until all activity on the file has quiesced before setting the file DISABLED or CLOSED. CICS then returns control to the application that is issuing this command. WAIT is the default.

Note: Closing a file using the FORCE option causes tasks of any current users of the file to be terminated immediately by the CICS task FORCEPURGE mechanism. Data integrity is not guaranteed with this mechanism. In some extreme cases (for example, if an error occurs during backout processing), CICS might terminate abnormally. For this reason, closing files using the FORCE option should be restricted to exceptional circumstances.

| **CFDTPOOL(***data-area***) (CFDT only)**
| specifies the name of the pool in which the coupling facility data table resides. You can specify the CFDT pool name for a file that does not currently refer to a coupling facility data table, but which could be switched to use a coupling facility data table at a later date.

DELETE(*cvda*) (VSAM only)
specifies whether records can be deleted from the file. CVDA values are:

DELETABLE
Records are deletable from the file.

NOTDELETABLE
Records are not deletable from the file.

DISPOSITION(*cvda*)
specifies the disposition for this file. When you issue a SET FILE DISPOSITION command, you override the current DISPOSITION value, which can have been taken from the installed file definition, or from any JCL statement for this file, if the file has been opened. CVDA values are:

OLD The disposition value is to be OLD.

SHARE
The disposition value is to be SHARE.

DSNAME(*data-value*)
specifies the data set name of the data set associated with this file, as defined to the access method and the operating system. The name can be up to 44 characters long. If no JCL statement exists for this file when it is opened, the open is preceded by a dynamic allocation of the file using this data set name. If there is a JCL statement, it takes precedence over the data set name specified this option.

If the file is associated with a coupling facility data table, DSNAME specifies the name of the source data set from which the table is loaded when the file definition specifies LOAD(YES).

Note: When you add a data set name to a file definition for a coupling facility data table, LSR pool size calculations may be involved when the file is opened. This occurs when the file refers to an LSRPOOL that CICS builds using default values, and the first data set using the LSR pool is opened to load the table. CICS issues message DFHFC0208 indicating that a delay might occur while the LSR pool size is being calculated. If you specify a data set name on a file that refers to an LSR pool that is already built using default values, the data set will not have been included in the LSR pool calculation. This means the existing LSR pool may not be adequate for the new data set. To resolve any problems associated with an LSR pool used in this way, you can close all files that reference the pool, which causes CICS to discard the pool and rebuild it using new calculations the next time a file is opened that refers to the pool. Alternativley, define the LSR pool explicitly specifying the appropriate values. See the *CICS Resource Definition Guide* for information about defining LSR pools.

With the SET FILE command, you can dissociate the file from any DSNAME by supplying a DSNAME value that begins with a null character (hexadecimal zeros).

EMPTY
is equivalent to EXEC CICS SET FILE EMPTYSTATUS(EMPTYREQ). It is supported for compatibility reasons only.

EMPTYSTATUS(*cvda*) (VSAM only)
specifies whether the data set is to be emptied when a file that references it is next opened. This is valid only for data sets that are defined as reusable, and that are accessed in either LSR or NSR mode. (EMPTYSTATUS does not apply if a file is opened in RLS mode.) CVDA values are:

EMPTYREQ
If the data set is defined as reusable, it is set to empty the next time a file that references it is opened in non-RLS mode.

Notes:

1. If you specify EMPTYREQ for a nonreuseable data set, CICS accepts it, but a subsequent attempt to open the file fails.
2. If you specify EMPTYREQ for a file defined with RLSACESS(YES), CICS accepts it, but the option does not have any effect unless the file is subsequently opened in non-RLS mode.
3. If you specify EMPTYREQ for a file that refers to a coupling facility data table that requires preloading from a data set and is specified with RLSACCESS(NO), and opening the file triggers the table load, the data set is set to empty.
4. If you specify EMPTYREQ for a file that refers to a coupling facility data table that does not require loading from a source data set, the option is ignored.
5. If you specify EMPTYREQ for a file that refers to a coupling facility data table that is already loaded from a source data set, the option is ignored.

NOEMPTYREQ
The data set has been defined as reusable but is not set empty the next time a file that references it is opened.Specify NOEMPTYREQ for a coupling facility data table.

ENABLESTATUS(*cvda*)
specifies whether application programs can access the file. CVDA values are:

DISABLED
The file is unavailable for access by application programs.

ENABLED
The file is available for access by application programs.

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 on this file are to be under exclusive control.

NOEXCTL
Records on this file are not to be under exclusive control.

FILE(*data-value***)**
specifies the 8-character file name defined in the file control table (FCT).

KEYLENGTH(*data-value***) (CFDT only)**
specifies, as a fullword binary value, the key length of records in a coupling facility data table. To set a keylength, specify a value in the range 1 through 16. To clear a keylength (set it to null values), specify KEYLENGTH(0).

You can specify the key length for a file that does not currently refer to a coupling facility data table, but which could be switched to use a coupling facility data table at a later date.

LOADTYPE(*data-value***) (CFDT only)**
specifies whether the coupling facility data table associated with the file requires pre-loading from a source data set. CVDA values are:

LOAD
The coupling facility data table requires loading from a source data set before it is fully usable; the transactions that use this coupling facility data table rely on it containing the records from the specified source data set.

NOLOAD
The coupling facility data table does not require loading from a source data set; it is fully usable as soon as it is created, and is populated by the transactions that use it.

You can specify the load type for a file that does not currently refer to a coupling facility data table, but which might be switched to use a coupling facility data table at a later date.

LSRPOOLID(*data-value***) (VSAM only)**
specifies, as a fullword binary value, the number of the LSR pool associated with this file. LSR pool IDs are in the range 1–8.

If the file is not to share buffers, set this value to 0.

For a CICS-maintained or user-maintained data table, the value must be 1 or greater. Both these types of CICS shared data table must use LSR access mode (unless the file is defined to be opened in RLS access mode).

For a coupling facility data table, you can set this value to 0.

MAXNUMRECS(*data-value***)**
specifies, as a fullword binary value, the maximum number of records the data table for this file can hold. Using this parameter enables you to control the use of storage.

For any type of table, if you want to set a limit, specify a value in the range 1 to 99 999 999. If you do not want any limit to apply, specify MAXNUMRECS(0), which CICS interprets as no limit, and sets internally to the maximum positive fullword value (+2147483647 or X'7FFFFFFF').

To specify MAXNUMRECS for a recoverable coupling facility data table, use a value that is between 5 and 10% more than the maximum number of records that the table is expected to contain. This allows for additional records that might be created internally for processing recoverable requests. The margin to be left for this internal processing depends on the level of use of the coupling facility data table, and the nature of that use. An effect of this is that the NOSPACE condition (with a RESP2 value of 102) can be raised on a WRITE or REWRITE request to a recoverable coupling facility data table that apparently has fewer records than the MAXNUMRECS limit specifies.

OPENSTATUS(*cvda***)**
specifies whether the file is to be open or closed. CVDA values are:

CLOSED
The file is to be closed.

The close request is deferred until all UOWs that hold repeatable read locks reach their syncpoint.

Note: A coupling facility data table remains in existence (in the coupling facility) after the file is closed, unlike a user-maintained data table, which ceases to exist when the file in the file-owning region is closed. Closing a file for a coupling facility data table does not prevent it being accessed through another file or by other CICS regions.

You can use the MVS MODIFY command to issue CFDT server commands that:

- Set the table unavailable (MODIFY *server-name*,SET TABLE=*tablename*,AVAILABLE=NO) so that no other files can issue opens against it
- Delete the table from the coupling facility (MODIFY *server-name*,DELETE TABLE=*table-name*) if you do not want

it to exist after the last file using it has been closed.

OPEN
The file is to be opened.

For a coupling facility data table, open processing causes the coupling facility data table server to create the table if it does not exist when CICS processes the open request. If the installed file definition specifies the name of a source data set, the coupling facility data table is created by loading the data from the source data set.

If a SET FILE(*filename*) OPEN command refers to a file that specifies LOAD(YES), but which does not name the source data set, the CFDT can be created and loaded only by opening a file that defines the source data set name.

If a recoverable data set is to be closed, the task the task issuing the close must commit any prior changes to that data set, or the request is rejected by file control.

READ(*cvda***)**
specifies whether records can be read from the file. CVDA values are:

NOTREADABLE
Records are not to be readable from the file.

READABLE
Records are to be readable from the file.

READINTEG(*cvda***)**
specifies the default level of read integrity for the file. CVDA values are:

CONSISTENT
Consistent read integrity is required for this file.

REPEATABLE
Repeatable read integrity is required for this file.

UNCOMMITTED
No read integrity is required for this file.

Note: These default read integrity values are used only when the file read request does not specify read integrity options explicitly on the EXEC CICS command.

CICS ignores a READINTEG option specified for a coupling facility data table.

RECORDSIZE(*data-area***) (CFDT only)**
specifies, as a fullword binary value, the maximum record size for a coupling facility data table in the range 1 through 32767.

You can specify the record size for a file that does not currently refer to a coupling facility data table, but which could be switched to use a coupling facility data table at a later date. Specify a record size of zero to remove a previously defined value.

RLSACCESS(*cvda***)**
specifies whether the file is to be accessed in RLS mode. The file must be closed, and either disabled or unenabled, to change the access mode to RLS access or to non-RLS access.

The non-RLS mode becomes either LSR or NSR, depending on the value specified for LSRPOOLID in the file resource definition.

CVDA values are:

NOTAPPLIC
The file is not eligible to be accessed in RLS mode because:

- It is a remote file, or
- It refers to a BDAM data set.

NOTRLS
The file is to be opened in LSR or NSR mode when it is next opened.

RLS The file is to opened in RLS access mode when it is next opened.

See *CICS Recovery and Restart Guide* for information about switching between RLS and non-RLS modes.

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 and CFDT only)**
specifies whether the file name specified on the FILE parameter represents a data table. CVDA values are:

CFTABLE
The file name refers to a coupling facility data table.

CICSTABLE
The file name represents a CICS-maintained data table.

NOTTABLE
The file name does not represent a data table.

USERTABLE
The file name represents a user-maintained data table.

TABLENAME(*data-area***) (CFDT only)**
specifies the 1- to 8-character name of the coupling facility data table to which this file refers. If TABLENAME is not specified, the table name defaults to the name of the file.

You can specify the table name for a file that does not currently refer to a coupling facility data table, but which could be switched to use a coupling facility data table at a later date.

UPDATE(*cvda***)**
specifies whether the file is read-only or read/write. CVDA values are:

NOTUPDATABLE
You can only read the records.

UPDATABLE
You can read, write, or delete the records.

| **UPDATEMODEL(*cvda*) (CFDT only)**
| specifies the type of update model to be used for a coupling facility data table. CVDA values are:

| CONTENTION
| The CFDT is to use the contention model, in which records are not locked when they are read for update, but an error is returned on a subsequent REWRITE or DELETE if the record has changed or been deleted since it was read for update.

| LOCKING
| The CFDT is to use the locking model, in which records are locked when they are read for update.

| You can specify the update model for a file that does not currently refer to a coupling facility data table, but which could be switched to use a coupling facility data table at a later date.

Conditions

FILENOTFOUND
RESP2 values:

18 The named file cannot be found.

INVREQ
RESP2 values:

1 The named file is REMOTE.

2 The named file is not CLOSED.

3 The named file is not DISABLED or UNENABLED.

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 is specified for a non-VSAM data set.

11 LSRPOOLID is not in the range 1–8, or the corresponding buffer is not defined.

12 READ has an invalid CVDA value.

13 STRINGS value is not in the range 1–255, or this is not a VSAM file.

14 UPDATE has an invalid CVDA value.

16 OPENSTATUS has an invalid CVDA value.

17 ENABLESTATUS has an invalid CVDA value.

19 DELETE has been specified for a non-VSAM file.

20 EMPTYSTATUS has been specified for a non-VSAM file.

21 CLOSED or DISABLED has been specified, but this transaction has an incomplete request against the file.

22 ENABLED was specified for a file that is currently DISABLING or UNENABLING.

23 EXCLUSIVE has an invalid CVDA value.

24 EXCLUSIVE has been specified for a non-BDAM file.

28 OPEN, CLOSE, ENABLE, or DISABLE has been specified but an exit program running at exit point XFCSREQ instructed CICS not to carry out the command.

29 TABLE has an invalid CVDA value.

30 MAXNUMRECS value is out of range.

31 The TABLE option is not valid for a BDAM file (must be VSAM for a data table).

32 The TABLE option is not valid for a file defined with the REUSE option.

33 The TABLE option is invalid for a file defined as UNBLOCKED.

34 The MAXNUMRECS option is invalid for a BDAM file (must be VSAM for a data table).

35 The MAXNUMRECS option is invalid for a file defined with the REUSE option.

36 The MAXNUMRECS option is invalid for a file defined as UNBLOCKED.

37 The TABLE option is invalid when LSRPOOL=0 is specified.

39 The USERTABLE option is invalid when record format is not variable.

40 CONSISTENT or REPEATABLE is specified for a file that is not accessed in RLS mode.

41 The DSNB cannot be disconnected, nor a new DSNB connected, for this file because the file has deferred work outstanding, for which there are retained or repeatable read locks. This means there is at least one shunted UOW, awaiting completion, that has made changes to this file.

42 The SET FILE request cannot be satisfied because the file has deferred work outstanding, for which there are retained or repeatable read locks. This means there is at least one shunted UOW, awaiting completion, that has made changes to this file. The only valid options when a file has deferred work pending are those that change the file state. File state changes are permitted because they may be required to enable the deferred work to be completed.

43 The file cannot be discarded because it has deferred work outstanding, for which there are retained or repeatable read locks. This means there is at least one shunted UOW, awaiting completion, that has made changes to this file.

44 A file open request cannot be satisfied because the file references a data set that has been marked as unavailable by a SET DSNAME UNAVAILABLE command.

45 A file open request cannot be satisfied because the file references an RLS-mode data set that has been quiesced by a SET DSNAME QUIESCED command.

46 A file open request cannot be satisfied because the file references an RLS-mode data set that is being copied by a DFSMSdss-initiated non-BWO backup.

47 A file open request cannot be satisfied because the file references an RLS-mode data set that is in the process of quiescing by a SET DSNAME QUIESCED command.

48 A file open request cannot be satisfied because the file references a data set for which its ICF Catalog entry indicates that a recovery is pending, or is in progress (for example, a CICSVR job is running).

49 An invalid CVDA is specified for the READINTEG option.

50 An attempt has been made to open an RLS file when RLS is not supported, either because the level of VSAM does not support RLS or because RLS=NO has been specified during system initialization.

51 An invalid CVDA is specified for the RLSACCESS option.

52 An attempt has been made to specify RLS access for a BDAM data set.

53 An attempt has been made to specify a CICS-maintained data table for a file defined with RLS access.

54 A file open request cannot be satisfied because of one of the following reasons:

- the file is being opened in RLS mode and this region has other files open in non-RLS mode against the data set which it references.
- the file is being opened in non-RLS mode and this region has other files open in RLS mode against the data set which it references.
- the file is being opened in non-RLS mode and this region has unresolved RLS recovery work against the data set which it references.

55 LOADTYPE has an invalid CVDA value.

56 UPDATEMODEL has an invalid CVDA value.

57 EMPTYSTATUS has a CVDA value that is not allowed for a coupling facility data table. EMPTYSTATUS must be NOEMPTYREQ for a coupling facility data table.

58 CFDTPOOL is not specified for a file that refers to a coupling facility data table.

59 KEYLENGTH is not specified for a file that refers to a coupling facility data table, and which specifies LOAD=NO.

60 An invalid KEYLENGTH is specified. The KEYLENGTH must be in the range 1 through 16 for a coupling facility data table.

61 RECORDSIZE is not specified for a file that refers to a coupling facility data table that specifies LOAD=NO.

62 An invalid RECORDSIZE is specified. RECORDSIZE must be in the range 0 through 32767 bytes.

63 OPEN is specified for a file that refers to a coupling facility data table, but OPEN processing has failed because:

- The file attributes do not match those specified when the CFDT was created, or
- A keylength or recordsize has been specified that exceeds the maximum supported.

64 OPEN is specified for a file that refers to a coupling facility data table, but OPEN processing has failed because the server is not available.

65 An invalid CFDTPOOL name is specified.

66 An invalid TABLE name is specified.

67 An UPDATEMODEL of CONTENTION is specified for a recoverable coupling facility data table. The update model must be LOCKING for a coupling facility data table that is recoverable.

IOERR

RESP2 values:

***** OPEN has failed in VSAM, and the RESP2 field contains a VSAM response code.

***** CLOSE has failed in VSAM, and the RESP2 field contains a 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

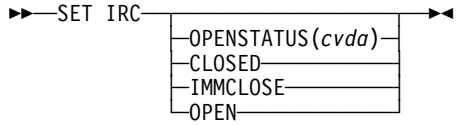
```
EXEC CICS SET FILE ('FILE12')
              WAIT
              CLOSED
              DISABLED
              DELETABLE
              LSRPOOLID(7)
              STRINGS(50)
EXEC CICS SET FILE ('FILE12')
              OPEN
              ENABLED
```

On the first command, the WAIT option tells CICS to allow all activity on FILE12 to quiesce before closing the file, and to return control to the issuing application only when this request has been started. When the file has been CLOSED, it is to be DISABLED. The records on FILE12 are then to be marked DELETABLE, LSRPOOL number 7 is to be associated with the file, and up to 50 concurrent operations are to be allowed.

The second of the two commands opens and then enables the file. Setting a file CLOSED and DISABLED makes the file eligible for deletion (DISCARD) or reinstallation by another task. Thus it is possible for another task to delete the file after the first SET command but before the second SET command.

SET IRC

Open or close interregion communication.



Conditions: INVREQ, IOERR, NOSTG, NOTAUTH

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 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.

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 log on to IRC failed.
14 An attempt to attach the node error transaction, CSNC, failed.
15 An error occurred closing IRC.

NOSTG
RESP2 values:

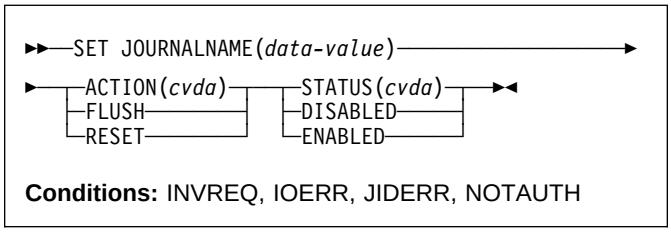
9 CICS storage is insufficient for the request.
10 MVS storage is insufficient (SVC block request rejected).
11 MVS storage is insufficient (SUBSYS block request rejected).

NOTAUTH
RESP2 values:

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

SET JOURNALNAME

Enable or disable a CICS user journal.



Conditions: INVREQ, IOERR, JIDERR, NOTAUTH

Description

The SET JOURNALNAME command allows you to enable or disable a CICS user journal.

SET JOURNALNAME has no effect on a journal that is being used as the forward recovery log or autojournal for a VSAM file until the next time the file is opened. It has no effect on the system log.

You can use SET JOURNALNAME for a journal name that is not currently known to CICS. CICS dynamically creates an entry for the specified journal and, if necessary, defines it to the MVS system logger using a matching JOURNALMODEL definition.

The ability to issue SET JOURNALNAME commands for journal names not known to CICS enables you to perform log stream connection processing before the corresponding journals are first referenced. For example, you could do this during a PLT program at initialization to avoid the delay that normally occurs at first reference.

Options

ACTION(*cvda*)
specifies the action you want CICS to take for the specified journal name. CVDA values are:

FLUSH
The log buffers are written out to the log stream, but the journal is not closed.

You can use this option to ensure that all current records are written out to the log stream before processing the stream using a batch utility.

In the case of autojournals and forward recovery logs, the FLUSH is forced if the file is open (the FLUSH does not wait until the next time the file is opened).

RESET
The journal is disconnected from its log stream, but can be reopened by a journal write.

Note: ACTION and STATUS are mutually exclusive options. If you specify ACTION, you cannot also specify STATUS.

JOURNALNAME(*data-value***)**
specifies the name of the journal.

To modify journals defined with a numeric identifier in the range 1–99, specify journal name DFHJ*nn*, where *nn* is the journal number.

You cannot specify DFHLOG or DFHSHUNT, because you are not allowed to modify the status of the system log.

STATUS(*cvda***)**
specifies the new status for the journal. The CVDA values are:

DISABLED
The journal is flushed then disabled. It cannot be used again until it is reenabled by the STATUS(ENABLED) or ACTION(RESET) options on a SET JOURNALNAME command.

ENABLED
The journal is open and is available for use.

Note: STATUS and ACTION are mutually exclusive options. If you specify STATUS, you cannot also specify ACTION.

Conditions

INVREQ
RESP2 values:

2 The request is invalid.
3 The system log cannot be changed.
4 The ACTION option has an invalid CVDA value.
5 The STATUS option has an invalid CVDA value.
n The ACTION option specifies FLUSH or RESET for a journal that is not currently connected to a log stream.

IOERR
RESP2 values:

6 The log stream associated with the journal name cannot be connected to, or the journal cannot be opened, or an unrecoverable error has occurred during the flushing of the log buffer to the log stream.

JIDERR
RESP2 values:

1 The specified journal name was not found.
2 An error occurred during an attempt to define the log stream associated with the journal name, or the journal name has been incorrectly specified.
3 The specified journal name refers to a DASD-only log stream to which a CICS region in another MVS image is currently connected.

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 JOURNALNUM

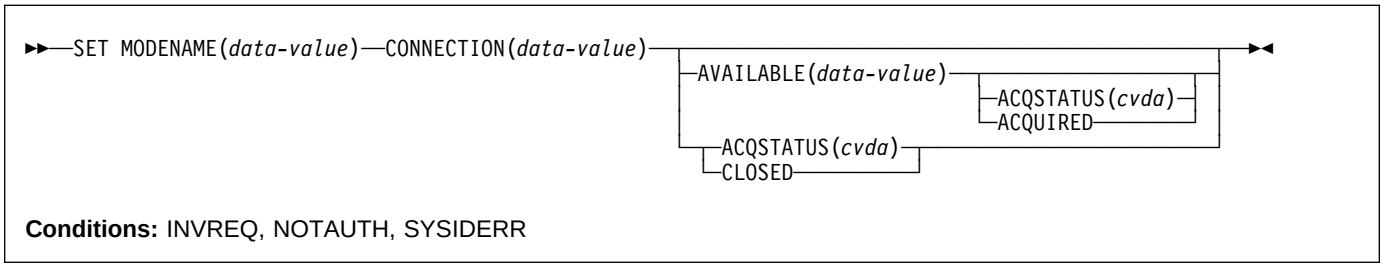
This command is supported in releases of CICS earlier than CICS Transaction Server for OS/390 for changing the OPENSTATUS setting of a journal.

Description

For CICS Transaction Server for OS/390, this command is replaced by the SET JOURNALNAME command. All the options on the SET JOURNALNUM are obsolete, and the only run-time support provided by CICS for compatibility with earlier releases is to return the JIDERR exception condition. The translator translates the command, but issues a warning message.

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 an implicit syncpoint).

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 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 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:

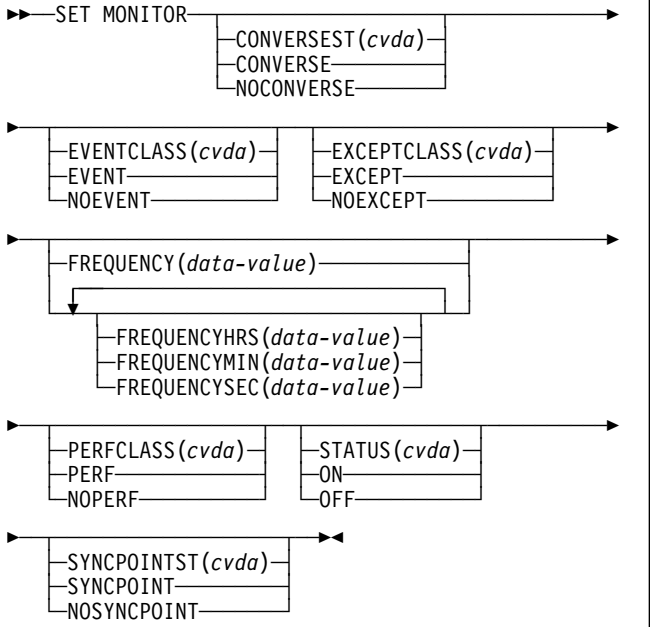
100 The user associated with the issuing task is not authorized to use this command.

SYSIDERR
RESP2 values:

1 The connection cannot be found.
2 The MODENAME within the connection cannot be found.

SET MONITOR

Change CICS monitoring options.



Conditions: INVREQ, NOTAUTH

For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” 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, and so on)
- 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 *0hhmmss+*, 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:

- 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 is accumulated or written out, irrespective of the settings of the monitoring data classes.

ON Monitoring is to be active. Data is 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 unit of work (UOW) within tasks that contain multiple UOWs. A UOW 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 UOW begins immediately, except at end of task. When rollback occurs on a syncpoint, the UOW does not end. CVDA values are:

NOSYNCPOINT
Performance data is to be combined over all UOWs in a task.

SYNCPOINT
Performance data is to be recorded separately for each UOW.

Conditions

INVREQ
RESP2 values:

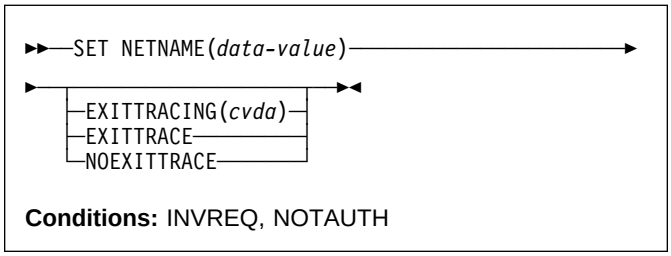
1 STATUS has an invalid CVDA value.
2 PERFCLASS has an invalid CVDA value.
3 EXCEPTCLASS has an invalid CVDA value.
4 EVENTCLASS has an invalid CVDA value.
5 CONVERSEST has an invalid CVDA value.
6 SYNCPOINTST has an invalid CVDA value.
7 The FREQUENCY value is invalid. (The hours exceed 24, minutes 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:

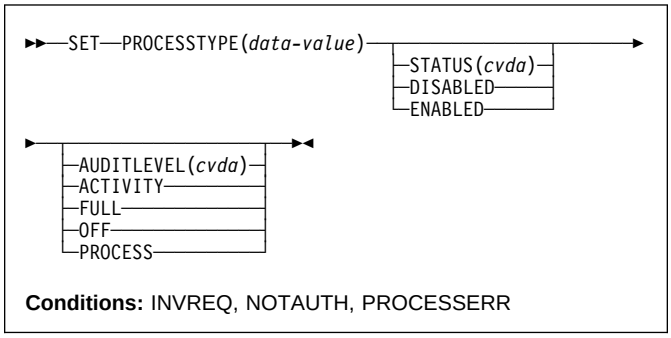
27 EXITTRACING has an invalid CVDA value.
29 The terminal is not a VTAM terminal.

NOTAUTH
RESP2 values:

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

SET PROCESSTYPE

Change the attributes of a CICS business transaction services (BTS) process-type.



Description

SET PROCESSTYPE allows you to change the current state of audit logging and the enablement status of PROCESSTYPE definitions installed on this CICS region.

Note: Process-types are defined in the process-type table (PTT). CICS uses the entries in this table to maintain its records of processes (and their constituent activities) on external data sets. If you are using BTS in a single CICS region, you can use the SET PROCESSTYPE command to modify your process-types. However, if you are using BTS in a sysplex, it is strongly recommended that you use CPSM to make such changes. This is because it is essential to keep resource definitions in step with each other, across the sysplex.

Options

AUDITLEVEL(cvda)
specifies the level of audit logging to be applied to processes of this type.

Note: If the AUDITLOG attribute of the installed PROCESSTYPE definition is not set to the name of a CICS journal, an error is returned if you try to specify any value other than OFF.

The CVDA values are:

ACTIVITY
Activity-level auditing. Audit records will be written from:

1. The process audit points
2. The activity primary audit points.

FULL
Full auditing. Audit records will be written from:

1. The process audit points
2. The activity primary *and* secondary audit points.

OFF No audit trail records will be written.

PROCESS
Process-level auditing. Audit records will be written from the process audit points only.

For details of the records that are written from the process, activity primary, and activity secondary audit points, see the *CICS Business Transaction Services* manual.

PROCESSTYPE(value)
specifies the 8-character name of a process-type defined in the process-type table (PTT), whose attributes are to be changed.

STATUS(cvda)
specifies whether new processes of this type can be created. The CVDA values are:

DISABLED
The installed definition of the process-type is disabled. New processes of this type cannot be defined.

ENABLED
The installed definition of the process-type is enabled. New processes of this type can be defined.

Conditions

INVREQ
RESP2 values:

2 The process-type is not disabled, and therefore cannot be enabled.
3 You have specified an invalid CVDA value on the AUDITLEVEL option.
5 You have specified an invalid CVDA value on the STATUS option.
6 You have specified a value of FULL, PROCESS, or ACTIVITY on the AUDITLEVEL option, but the AUDITLOG attribute of the PROCESSTYPE definition does not specify an audit log.

NOTAUTH
RESP2 values:

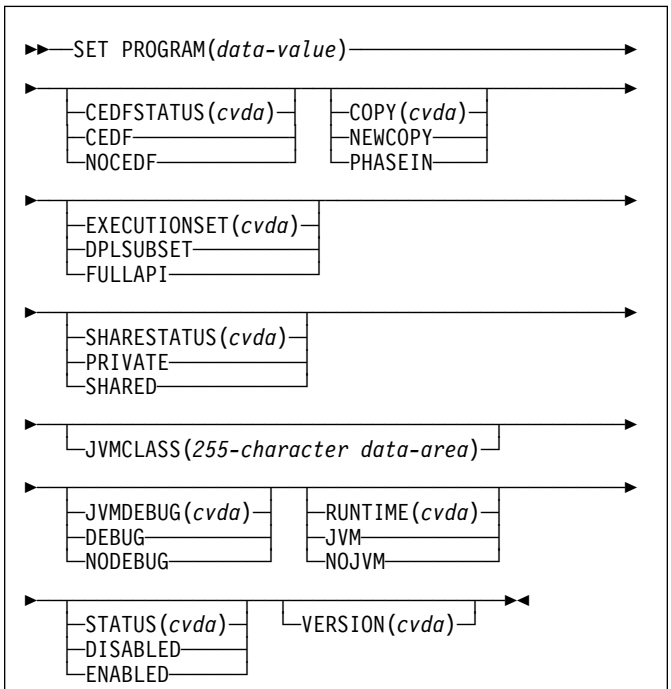
100 The user associated with the issuing task is not authorized to use this command.

PROCESSERR
RESP2 values:

1 The process-type named in the PROCESSTYPE option is not defined in the process-type table (PTT).

SET PROGRAM

Change a PROGRAM, MAPSET, or PARTITIONSET definition.



Conditions: INVREQ, IOERROR, NOTAUTH, PGMIDERR

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.

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. Any value specified for JVM programs is ignored.

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.

The EXECUTIONSET attribute applies only:

- to programs which are being linked to, and not to those which are the first to be given control by a transaction.
- when the REMOTESYSTEM name is the same name as the local CICS region. Its purpose is to test programs in a local CICS environment as if they were running as DPL programs.

JVMCLASS(*255–character data-area***) (JVM programs only)**
specifies the name of the main class in the Java program to be given control by the JVM. If you specify JVM in the RUNTIME option, you should specify a JVMCLASS value. If you specify NOJVM in the RUNTIME option, any value in the JVMCLASS option will be ignored when the program is executed.

JVMDEBUG(*cvda***) (JVM programs only)**
specifies whether or not the JVM is to operate in debugging mode for this program. CVDA values are:

DEBUG
The JVM is to operate in debugging mode for this program. The JVM operates in debug mode, a password is output to `stdout` to be used to attach a remote JDB debugging tool.

NODEBUG
The JVM is not to operate in debugging mode for this program.

PROGRAM(*data-value***)**
specifies the 8-character name of the program, map set, or partition set definition to be changed.

RUNTIME(*cvda***) (JVM programs only)**
specifies whether the program is to run under the control of a JVM. CVDA values are:

JVM The program is to run under the control of a JVM. You should specify a JVMCLASS value.

NOJVM
The program is not to run under the control of a JVM. Any value in the JVMCLASS option is ignored and the runtime environment of the changed program is unknown until the program is next loaded by CICS, at which point its language, and hence whether it will run with LE370, will be determined.

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 is loaded as if SHARESTATUS were PRIVATE.

You cannot specify SHARESTATUS for a remote program. Any value specified for JVM programs is ignored.

STATUS(*cvda***)**
specifies whether the module is to be available for use. CVDA values are:

DISABLED
The module is to be unavailable. CICS programs (beginning with 'DFH') cannot be disabled.

ENABLED
The module is to be available.

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. This value is always returned for JVM programs.

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.

21 JVMDEBUG has an invalid CVDA value.

22 RUNTIME has an invalid CVDA value.

23 JVM was specified but no JVMCLASS has been supplied.

24 DEBUG was specified but RUNTIME was not set to JVM.

25 JVMCLASS contains embeded blanks or null (x'00') characters.

IOERR
RESP2 values:

8 Either the COPY option or the RUNTIME(NOJVM) 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.

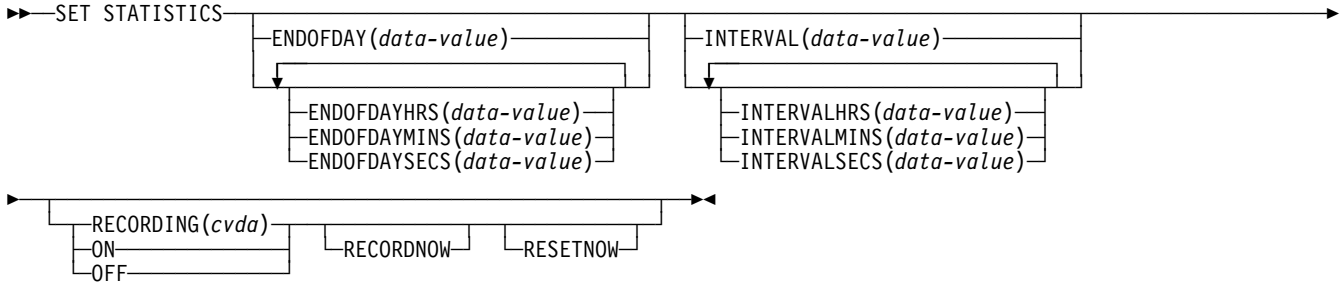
Examples

```
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 loads the new copy and makes it the current one.

SET STATISTICS

Change the recording of CICS statistics.



Conditions: INVREQ, NOTAUTH

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

Description

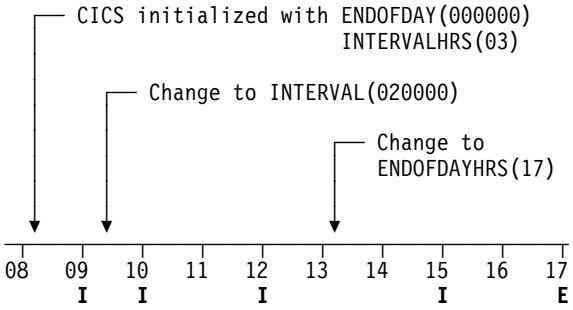
The SET STATISTICS command allows you to change values that control the recording of CICS statistics and to reset the counts.

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. 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 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 *0hhmmss+*, 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 *0hhmmss+*.

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 *0hhmmss+*. 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 interval statistics are to be recorded. (Statistics are always accumulated, and end-of-day, unsolicited, and requested statistics always recorded, regardless 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 237, or by a CEMT PERFORM STATISTICS transaction.)

CVDA values are:

OFF Recording is off.

ON Recording is on.

RECORDNOW
specifies that the current statistics are to be to 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 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:

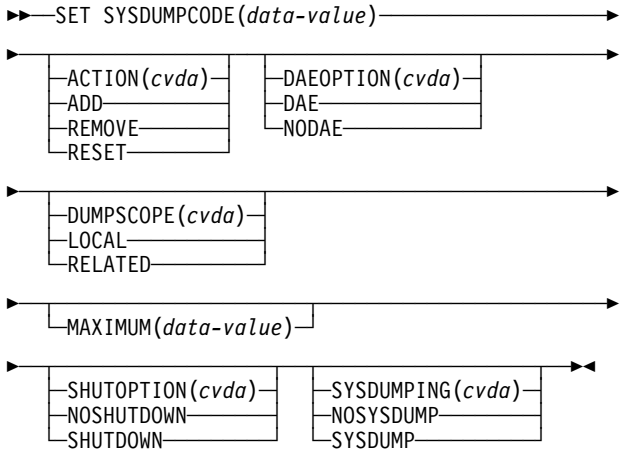
1 The INTERVAL value is out of range.
2 The ENDOFDAY value is out of range.
3 RECORDING has an invalid CVDA value.
4 The INTERVALHRS value is out of range.
5 The INTERVALMINS value is out of range.
6 The INTERVALSECS value is out of range.
7 More than one of the interval values has been used and the 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.



Conditions: DUPREC, INVREQ, IOERR, NOSPACE, NOTAUTH, NOTFND

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 the 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 an initial or 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 must remove the dump code from the table and then add it back.

For information about system dumps, see the *CICS Problem Determination Guide*.

Options

ACTION(*cvda*)
specifies the action 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 does 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 *OS/390 MVS Diagnosis: Tools and Service Aids* manual.

When SET SYSDUMPCODE ADD is specified, if you do not also specify DAEOPTION, it defaults 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 causes 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.

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.

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.

Conditions

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.
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:

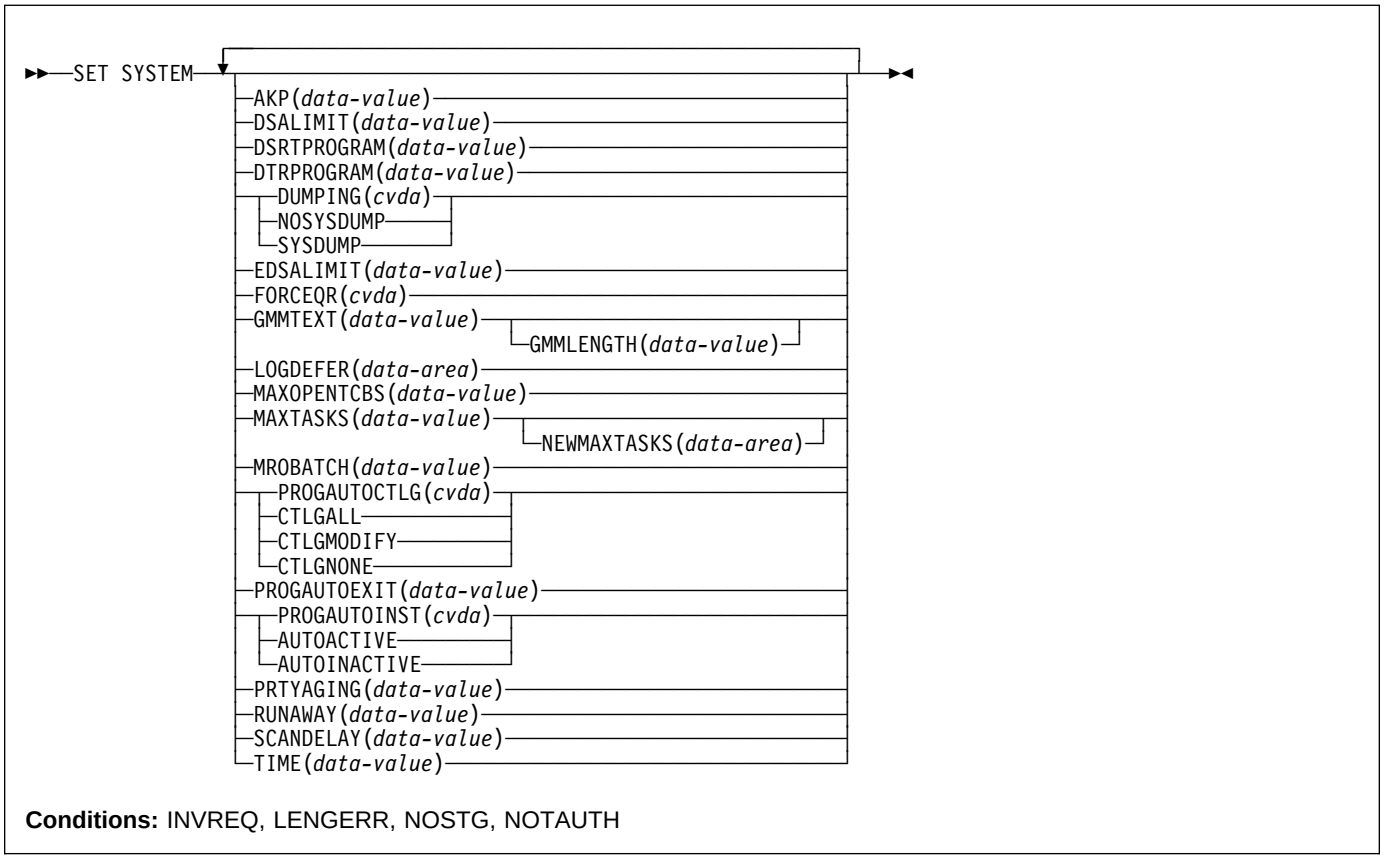
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The dump code cannot be found.

SET SYSTEM

Change CICS system option values.



Conditions: INVREQ, LENGERR, NOSTG, NOTAUTH

\

Notes:

1. For more information about the use of CVDAs, see “CICS-value data areas (CVDAs)” 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. 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 System Definition Guide*. System initialization parameters that correspond to those in this command have the same or similar names, except where noted. Table 2 on page 169 lists the exact correspondence.

Options

AKP(*data-value***)**
specifies, as a fullword binary value, the activity keypoint trigger value, which is the number of write requests to the CICS system log stream output buffer between the taking of keypoints. The number must be either zero, which turns off keypointing, or in the range 200–65535. If CICS was initialized without keypointing (that is, with the AKPFREQ system initialization parameter set to zero), the initial value can be overridden and a trigger value can be set.

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.

DSRTPROGRAM(*data-value***)**
specifies the 8-character name of the distributed routing program.

DTRPROGRAM(*data-value***)**
specifies the 8-character name of the dynamic routing program.

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.

FORCEQR(*cvda***)**
specifies whether you want CICS to force all user application programs specified as CONCURRENCY(THREADSAFE) to run under the CICS QR TCB, as if they were specified as CONCURRENCY(QUASIRENT) programs.

This allows you, in a test environment, to run incompletely tested threadsafe application programs that have proved to be non-threadsafe.

FORCEQR applies to all programs defined as threadsafe that are not invoked as task-related user exits, global user exits, or user-replaceable modules.

CVDA values are:

FORCE
All user programs defined as threadsafe are to be forced to execute as quasi-reentrant programs.

NOFORCE
CICS is to honor the CONCURRENCY(THREADSAFE) attribute defined on program resource definitions, and invoke them under either the QR TCB or an open TCB.

The FORCEQR(FORCE|NOFORCE) option allows you to change dynamically the option specified by the FORCEQR system initialization parameter.

Specifying FORCEQR(FORCE) is not applied to currently invoked programs, and applies only to programs invoked for the first time after the change to the FORCEQR status.

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.

LOGDEFER(*data-area***)**
specifies, as a halfword binary value, the log deferral interval, which is the period of time used by CICS Log Manager when determining how long to delay a forced journal write request before invoking the MVS system logger. The value can be in the range 0-65535. See the *CICS System Definition Guide* for information about the LOGDEFER parameter and associated SIT parameter LGDFINT.

MAXOPENTCBS(*data-value***)**
specifies, as a fullword binary value, the maximum number of open TCBs that can exist concurrently in the CICS region. The value specified can be in the range 1 to 999.

If you reduce MAXOPENTCBS from its previously defined value, and the new value is less than the number of open TCBs currently allocated, CICS detaches TCBs to achieve the new limit only when they are freed by user tasks. Transactions are not abended to allow TCBs to be detached to achieve the new limit.

If there are tasks queued waiting for an open TCB, and you increase MAXOPENTCBS from its previously defined value, they will be resumed when CICS attaches new TCBs up to the new limit.

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 Transaction Server for OS/390 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 210).

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 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.

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.
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.
29 The LOGDEFER value is out of range.

LENGERR
RESP2 values:

20 The GMMLENGTH value is out of range.

NOSTG
RESP2 values:

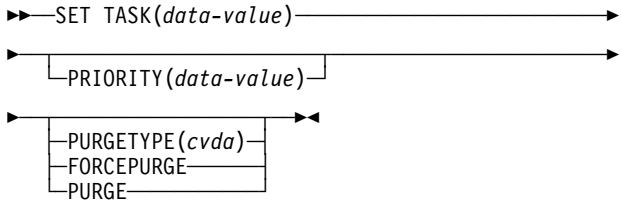
16 CICS reduced the value you requested for MAXTASKS because of storage constraints; see the NEWMAXTASKS option.

NOTAUTH
RESP2 values:

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

SET TASK

Purge a task or change its priority.



Conditions: INVREQ, NOTAUTH, TASKIDERR

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 it is 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:

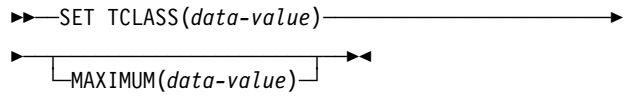
100 The user associated with the issuing task is not authorized to use this command.

TASKIDERR
RESP2 values:

1 The task cannot be found.
2 The task is protected by CICS and is not eligible for modification with this command.

SET TCLASS

Set the maximum number of tasks in a transaction class.



Conditions: INVREQ, NOTAUTH, TCIDERR

Description

The SET TCLASS command allows you to set the maximum number of tasks in a particular transaction class that 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 309, 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 309 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:

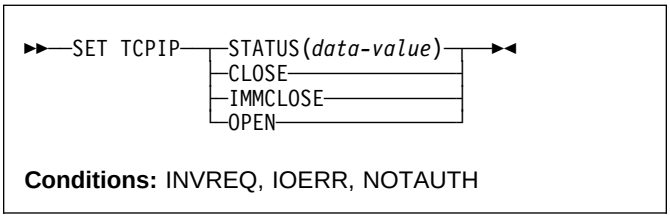
100 The user associated with the issuing task is not authorized to use this command.

TCIDERR
RESP2 values:

1 The transaction class cannot be found.

SET TCPIP

Modify CICS internal TCPIP support.



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

Description

SET TCPIP allows you to open or close CICS internal sockets support.

Options

STATUS(*cvda***)**
specifies whether TCPIP is to be enabled (that is, able to process new incoming work, and complete ongoing work), and if TCPIP support is to be disabled, how the disable should be done. CVDA values are:

OPEN CICS internal TCPIP support is to be opened.

CLOSED CICS internal sockets support is to be closed. If it is currently open, CICS is to quiesce all internal sockets activity and then close any sockets on which CICS is listening for incoming CICS Web Interface work. Tasks using CICS internal sockets are allowed to complete.

IMMCLOSE CICS internal sockets is to be closed. If it is currently enabled, CICS is to terminate abnormally any tasks using it and then close the socket on which CICS is listening for incoming work.

Conditions

INVREQ
RESP2 values:

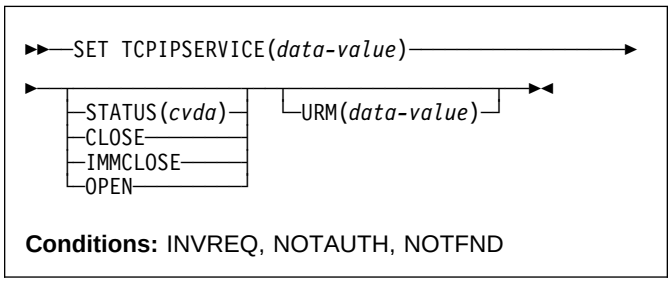
4 TCPIP not available (TCPIP=NO)
5 TCPIP already closed (for SET CLOSED|IMMCLOSE)
6 TCPIP already open (for SET OPEN)
11 STATUS has an invalid CVDA value.
12 The OPEN request did not complete because another task subsequently requested a CLOSE of CICS internal sockets support.

NOTAUTH
RESP2 values:

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

SET TCPIPSERVICE

Modify the status of a service using CICS internal TCPIP support.



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

Description

SET TCPIPSERVICE tells CICS to start or stop listening for incoming requests on the port associated with a service using CICS internal TCPIP support.

Note: This command has no effect on the sockets support provided by the TCPIP for OS/390 CICS Sockets Feature.

Options

STATUS(*cvda*)
Changes the status of the service. CVDA values are:

OPEN CICS internal TCPIP support is to start listening for incoming requests on the specified port.

CLOSED CICS internal sockets support is to stop listening for incoming work on the specified port.

IMMCLOSE CICS internal sockets support is to stop listening for incoming work on the specified port. If it is currently active, CICS is to terminate abnormally any tasks using it.

URM(*cvda*)
Specifies the 8-character name of the program to be used as the Service User-replaceable module. You can specify either an installation-specific program or the CICS-supplied default for the service. Some services may not allow this name to be changed.

Note: This program and any progams it invokes must be installed before they can be used in the program autoinstall process. You can do this either with explicit PROGRAM definitions or by autoinstall when some other autoinstall program is in force. Otherwise the program autoinstall process fails when it is next used, and CICS makes it inactive.

Conditions

INVREQ
RESP2 values:

4 TCPIP is not available (TCPIP=NO)
5 TCPIP status is closed
7 Port is in use
8 CICS is not authorized to use this port
9 TCPIPSERVICE not closed
10 Unknown IP address
11 Invalid value specified in an operand of the SET command, (the **URMname** is wrong).
12 Invalid status

NOTAUTH
RESP2 values:

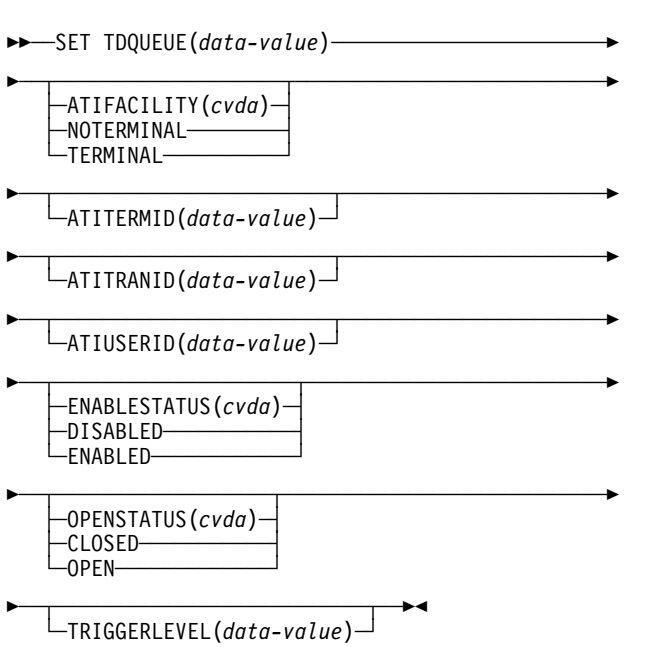
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

3 the named TCPIPSERVICE is not found

SET TDQUEUE

Change the attributes of a transient data queue.



Conditions: INVREQ, IOERR, NOTAUTH, QIDERR, USERIDERR

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 value specified on the TRIGGERLEVEL option. If the value 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.)

You cannot alter the following parameters unless the queue is fully disabled:

- ATIFACILITY
- ATITERMID
- ATITRANID.

To disable a transient data destination, the queue must not currently be in use. If it is in use, the queue enters a “disable pending” state. The last unit of work (UOW) to use the queue fully disables it. The parameters TRIGGERLEVEL, OPENSTATUS, and ENABLESTATUS can be altered regardless of whether the queue is enabled or disabled. The value of the ENABLESTATUS parameter cannot be altered while a queue is in a “disable pending” state.

A transient data queue cannot be disabled while it is in use, or while tasks are waiting to use it.

Indirect and remote queues can be disabled at any time because they have no concept of being “in use”.

If tasks are waiting to use an extrapartition queue, a physically recoverable queue, or a non-recoverable intrapartition queue, and an attempt is made to disable the queue, the queue enters a “disable pending” state. The last task to use the extrapartition queue fully disables it.

If an attempt is made to disable a logically recoverable intrapartition TD queue when there are UOWs enqueued upon it, the queue enters a “disable pending” state. The last UOW to obtain the enqueue fully disables the queue. If a UOW has updated a logically recoverable queue and suffers an in-doubt failure, the queue cannot be disabled until the in-doubt failure has been resolved.

If a UOW owns an enqueue on a queue that is in a “disable pending” state, it is allowed to continue making updates.

When a queue is in a “disable pending” state, no new tasks can alter the queue’s state or its contents. A disabled response is returned when a READQ, WRITEQ, or DELETEQ request is issued against a destination that is in a “disable pending” state.

Note: If a task updates a logically recoverable transient data queue, and attempts to disable the queue and alter an attribute of the queue (for example, ATITRANID) within the same UOW, the call fails. This is because the UOW is a user of the queue, and the queue enters a “disable pending” state. The SET operation on the queue attribute, in this case, ATITRANID, fails. The queue does not become fully disabled until the UOW commits or backs out at syncpoint. You are recommended to issue an EXEC CICS SYNCPOINT command before attempting to update the queue attribute (ATITRANID) using SET TDQUEUE.

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 the 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 abends 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 is 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 disable a queue that has suffered an in-doubt failure.

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 opened.

TDQUEUE(*data-value***)**
specifies the 4-character name of the transient data queue 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 it is zero, ATI does not occur. If it is not zero, when the queue reaches the TRIGGERLEVEL depth CICS creates a task to process it automatically. See also the ATIFACILITY, ATITERMINAL, ATITRANSID, 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.

9 OPENSTATUS was specified for an intrapartition queue.

10 ENABLESTATUS has an invalid CVDA value.

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.

18 SET not possible because the queue was not closed.

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.

30 Disabled pending condition.

31 SET not possible because the queue was not disabled.

35 SET not possible because the queue is in-doubt.

40 SET not possible because the queue is CXRF.

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 on 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 *OS/390 Security Server (RACF) Security Administrator's Guide*, SC28-1915

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:

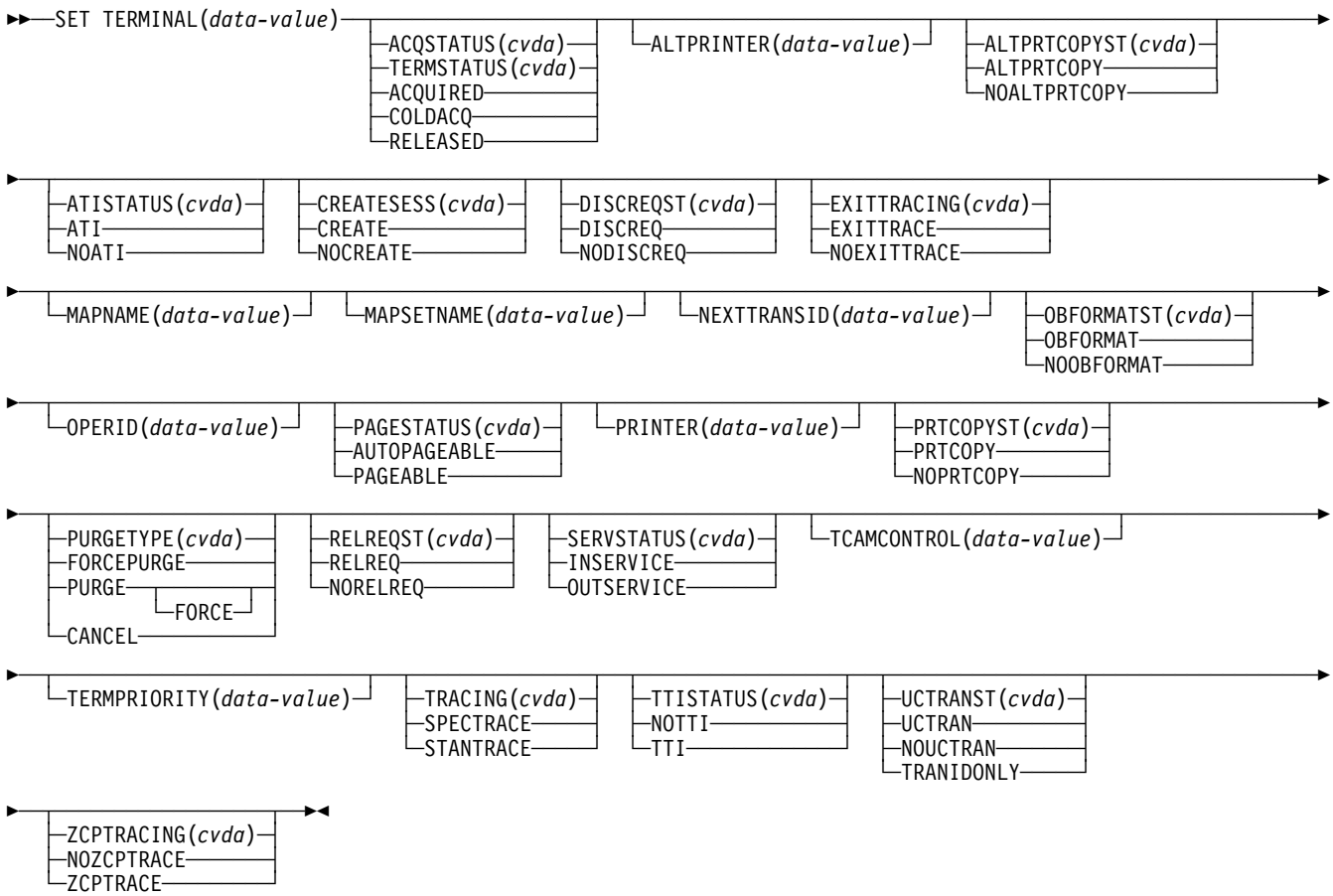
1 The queue cannot be found.

USERIDERR
RESP2 values:

28 The user named in ATIUSERID is not known to the ESM.

SET TERMINAL

Change some terminal attributes and cancel outstanding AIDs.



Conditions: INVREQ, NORMAL, NOTAUTH, TERMIDERR

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

Description

The SET TERMINAL command changes some of the values of a named terminal definition. It cannot be used for APPC sessions.

Only PURGETYPE, PURGE, or FORCEPURGE can be used for IRC sessions.

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 that 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 is shipped back to the TOR and intermediate systems. Any attempt to change any other attribute for a model or surrogate terminal results in INVREQ with RESP2=24.

Options

ACQSTATUS(*cvda***) (VTAM only)**
is retained only for compatibility purposes. You should use TERMSTATUS in new applications (see page 302).

ALTPRINTER(*data-value***)**
specifies the name of a 3270 printer for use as an alternative to the printer defined on the PRINTER option. The name can be up to four characters long.

Note: You cannot specify ALTPRINTER for a terminal that does not have a primary printer defined (on the PRINTER parameter).

See the *CICS Resource Definition Guide* for information about the PRINTER and ALTPRINTER parameters for defining primary and alternate printers for terminals.

Note: For VTAM terminals, in a transaction routing environment, this command does not take effect until the next flow across the link from the TOR to the AOR for the named terminal.

ALTPRTCOPYST(*cvda***)**
specifies the alternate printer copy status. This indicates whether CICS is to use the hardware COPY feature to satisfy a print request on the printer named on the ALTPRINTER parameter. CVDA values are:

ALTPRTCOPY
CICS is to use the hardware COPY feature to satisfy a print request on the alternate printer.

NOALTPRTCOPY
CICS is not to use the hardware COPY feature.

Note: You cannot specify ALTPRTCOPY for a terminal that does not have an alternate printer defined.

See the *CICS Resource Definition Guide* for information about the ALTPRINTCOPY parameter, which specifies the use of the hardware copy feature for the alternate printer on the terminal definition.

ATISTATUS(*cvda***)**
specifies whether the terminal can be used by transactions that are automatically initiated from within CICS or, if the terminal is an ISC session, by transactions that are using this session as an alternate facility to communicate with another system. CVDA values are:

ATI The terminal can be used by automatically initiated transactions.

NOATI
The terminal cannot be used by automatically initiated transactions.

A terminal cannot have both NOATI and NOTTI in its status.

CREATESESS(*cvda***) (VTAM only)**
specifies whether the terminal can be acquired automatically by ATI transactions. CVDA values are:

CREATE
The terminal can be acquired automatically.

NOCREATE
The terminal cannot be acquired automatically.

DISCREQST(*cvda***)**
specifies whether CICS is to honor a disconnect request from the terminal. CVDA values are:

DISCREQ
CICS honors a disconnect request for a VTAM device, and issues a VTAM CLSDST macro instruction to terminate the VTAM session with that logical unit.

It also means that CESF LOGOFF (or GOODNIGHT) from the terminal causes disconnection.

NODISCREQ
CICS does not honor a disconnect request for a VTAM device.

EXITTRACING(*cvda***)**
specifies whether the activity associated with the terminal exit program is to be traced. CVDA values are:

EXITTRACE
Exit program activity is to be traced.

NOEXITTRACE
Exit program activity is not to be traced.

MAPNAME(*data-area***)**
specifies the 7-character name of the map that is to be saved (stored) 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 region when the currently executing transaction terminates, unless the map name is superseded by a subsequent SEND MAP command. You can use the MAPNAME option to restore a map name that was returned to the application program 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-area***)**
specifies the 8-character name of the mapset that is to be saved 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 region 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 that was returned to the application program 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 next transaction identifier for the specified terminal. The identifier can be up to 4 characters long. If you specify the NEXTTRANSID parameter as blanks

(X'40404040'), CICS sets the next transaction identifier to nulls, meaning there is no NEXTTRANSID defined for the terminal.

Changes are permitted to a remote TCTTE, but the change is not shipped back to the TOR.

Note: NEXTTRANSID cannot be set if a transaction has been defined for this terminal.

OBFORMATST(*cvda***)**
specifies whether the device supports outboard formatting. See the TYPETERM definition in the *CICS Resource Definition Guide* for details of the types of device that support outboard formatting. CVDA values are:

NOOBFORMAT
The device does not support outboard formatting.

OBFORMAT
The device supports outboard formatting.

Note: OBFORMATST cannot be specified for a console or 3790.

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 are to be written. CVDA values are:

AUTOPAGEABLE
Pages, after the first in a series, are to be written to the terminal automatically.

PAGEABLE
Pages, after the first in a series, are to be written to the terminal on request from the operator.

PRINTER(*data-value***)**
specifies the name of the primary printer CICS is to use in response to a print request (either an ISSUE PRINT command, or a PRINT request from an operator pressing a program access (PA) key). The name can be up to 4 characters long. See the *CICS Resource Definition Guide* for information about specifying 3270-type printers.

| **Note:** For VTAM terminals, in a transaction routing
| environment, this command does 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 on the printer named on the PRINTER parameter. CVDA values are:

NOPRTCOPY
CICS is not to use the hardware COPY feature.

PRTCOPY
CICS is to use the hardware COPY feature to satisfy a print request on the primary printer.

Note: You cannot specify PRTCOPY for a terminal that does not have a printer defined.

See the *CICS Resource Definition Guide* for information about the PRINTCOPY parameter, which specifies the use of the hardware copy feature for the primary printer on the terminal definition.

PURGETYPE(*cvda***)**
specifies whether transactions running with the named terminal can be purged. CVDA values are:

CANCEL
AIDs queuing for the specified terminal are to be canceled. AIDs representing scheduled and allocated requests waiting in the local CICS system for the specified terminal are canceled. However, CRSR AIDs and TD AIDs with an associated triggered task already started are not canceled.

When a canceled scheduled request is found to have a precursor in a remote CICS system, this remote AID is canceled asynchronously. Message DFHTF0100 is written to CSMT to indicate how many AIDs have been deleted for the terminal and how many remain.

FORCEPURGE
Transactions are to be purged immediately. This can lead to unpredictable results and should be used only in exceptional circumstances.

PURGE
The transactions can be terminated only if system and data integrity can be maintained. A transaction is to be purged if its definition specifies SPURGE=NO.

FORCEPURGE replaces PURGE FORCE, which is retained only for compatibility purposes. You should use FORCEPURGE in new applications.

PURGETYPE cannot be specified for non-VTAM terminals.

RELREQST(*cvda***)**
specifies the status for releasing the logical unit. CVDA values are:

NORELREQ
CICS is not to release the logical unit upon request by another VTAM application program.

RELREQ
CICS is to release the logical unit, if the logical unit is not currently busy running a transaction.

SERVSTATUS(*cvda***)**
specifies whether the terminal is to be in- or out-of-service. CVDA values are:

INSERVICE
CICS is to set the terminal in-service and available for use.

OUTSERVICE
CICS is to set the terminal out-of-service, and not available for transactions. Unless you specify PURGE or FORCEPURGE, any current transaction is allowed to terminate normally, but no further transactions are allowed to use the terminal.

If the execution diagnostic facility (EDF) is in use at the specified terminal, EDF stops immediately, because it is a sequence of separate transactions, while the transaction that is being tested under EDF is allowed to complete.

If you set a VTAM terminal to OUTSERVICE, it is also RELEASED and the operator is signed off, either immediately or when the current transaction has terminated. You cannot therefore set the terminal associated with the executing transaction to OUTSERVICE, unless it is a printer.

TCAMCONTROL(*data-value***)**
specifies a 1-character hexadecimal value in the TCAM control byte indicating which segment of the message is to be 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.

TERMINAL(*data-value***)**
specifies the 4-character terminal name.

Note: As a result of the operation of the XICTENF and XALTENF global user exits, it is possible for SCHEDULE requests to be queued for a terminal that is not yet defined to the local CICS system. You can use the SET TERMINAL(*data-value*) CANCEL command to remove these requests.

TERMPRIORITY(*data-value***)**
specifies, as a fullword binary value, the priority required for the terminal, relative to other terminals, in the range 0–255.

TERMSTATUS(*cvda***) (VTAM only)**
specifies the session status for the logical unit represented by this terminal. CVDA values are:

ACQUIRED
CICS is to acquire a session with the logical unit represented by this terminal.

COLDACQ
CICS is to acquire a session with the logical unit represented by this terminal where no resynchronization is required.

RELEASED
CICS is to terminate the session. This happens immediately if you also specify the PURGE option, otherwise the session is terminated when the current active transaction finishes.

TRACING(*cvda***)**
specifies the required tracing activity associated with the terminal. CVDA values are:

SPECTRACE
Special tracing is to be used.

STANTRACE
Standard tracing is to be used.

Changes are permitted to a remote TCTTE, but the change is not shipped back to the TOR.

TTISTATUS(*cvda***)**
specifies whether this terminal can be used by the transactions that are initiated from this terminal. CVDA values are:

NOTTI
This terminal cannot be used by transactions initiated from it.

TTI This terminal can be used by transactions initiated from it.

A terminal cannot be defined with both NOATI and NOTTI.

UCTRANST(*cvda***)**
specifies whether the uppercase translate option is to be set for transactions associated with this terminal. Note that there is also an UCTRAN option on the profile definition. See Table 5 on page 303 for information on how the UCTRAN options on the terminal and transaction profiles interact.

| 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 is shipped back to the TOR and intermediate systems. Attempting to change any other attribute for a model or surrogate terminal results in INVREQ with RESP2=24.

This command may be used to set the uppercase 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 is raised with a RESP2 value of 24. The uppercase translation option is also changed in the terminal-owning region and any intermediate region in a daisy-chaining setup. CVDA values are:

NOUCTRAN
: CICS is not to perform uppercase translation on input from this terminal (unless specified otherwise on the profile for individual transactions).

TRANIDONLY
: CICS is to perform uppercase translation on the transaction id only on input from this terminal.

UCTRAN
: CICS is to perform uppercase translation on input from this terminal.

Table 5. The effect of the UCTRAN parameters

Profile	Terminal (TYPETERM)		
	UCTRAN (YES)	**UCTRAN (NO)**	**UCTRAN (TRANID)**
UCTRAN (YES)	Tranid: Yes Data: Yes	Tranid: No Data: Yes	Tranid: Yes Data: Yes
UCTRAN (NO)	Tranid: Yes Data: Yes	Tranid: No Data: No	Tranid: Yes Data: No

Note: This table shows which portion of the terminal input is translated (transaction id and/or data) according to the setting of the UCTRAN on the PROFILE and TYPETERM resource definitions.

ZCPTRACING(*cvda***)**
specifies the required tracing activity associated with the VTAM control component of CICS. CVDA values are:

NOZCPTRACE
: VTAM ZCP tracing is not to be carried out.

ZCPTRACE
: VTAM ZCP tracing is to be carried out.

Conditions

INVREQ
RESP2 values:

1 TERMSTATUS or ACQSTATUS was specified for an IRC session or non-VTAM terminal.

2 TERMSTATUS or ACQSTATUS has an invalid CVDA value.

4 ATISTATUS has an invalid CVDA value.

5 ATISTATUS change 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 Trying to put the issuing terminal OUTSERVICE.

13 SERVSTATUS has an invalid CVDA value.

15 TERMPRIORITY value not in range 0–255.

17 NOTTI cannot be specified for the issuing terminal.

18 TTISTATUS has an invalid CVDA value.

21 PURGETYPE has an invalid CVDA value.

22 TRACING has an invalid CVDA value.

24 Invalid option requested for a remote terminal.

25 ACQUIRED specified, but terminal is not in service.

26 PURGE specified, but target task has SPURGE=NO on its associated transaction definition.

27 EXITTRACING has an invalid CVDA value.

28 ZCPTRACING has an invalid CVDA value.

29 EXITTRACING or ZCPTRACING specified for a non-VTAM terminal (or VTAM not installed).

31 This is a remote terminal with no associated surrogate.

33 SET TERMINAL is not valid for an LU6.2 (APPC) session.

34 A permanent transaction has been defined for this terminal (TRANSACTION operand in TERMINAL definition).

35 Attempt made to change TCAM CONTROL on non-TCAM terminal.

36 Invalid value supplied for TCAM CONTROL.

37 Preset signon failed, terminal remains OUTSERVICE.

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 Options would result in the invalid combination of the alternate printer copy status being set without an alternate printer defined.

45 Options would result in the invalid combination of the alternate printer being defined without a primary printer defined.

46 OBFORMATST is specified for a console or 3790.

48 Options would result in the invalid combination of the printer copy status being set without a primary printer defined.

50 VTAM not available for VTAM terminal.

51 PRINTER and ALTPRINTER option specified for a terminal that is not VTAM 3270 or 3270 compatibility mode.

52 PRTCOPYST or ALTPRTCOPST option specified for a terminal that is not VTAM 3270 or 3270 compatibility mode.

54 Option other than PURGETYPE specified for IRC session.

57 Other SET parameters were included with the CANCEL option.

NORMAL
RESP2 values:

53 Purge deferred.
58 AIDs are successfully canceled.
59 No AIDs are canceled.
60 MAPNAME or MAPSETNAME specified, but the terminal is not of a type supported by BMS.

NOTAUTH
RESP2 values:

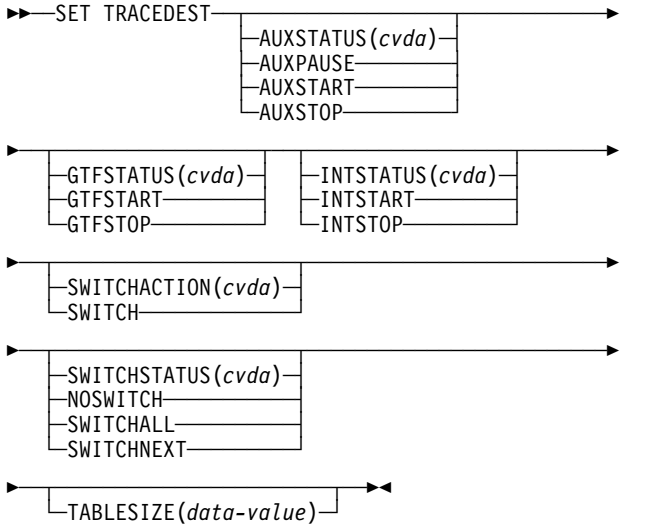
100 The user associated with the issuing task is not authorized to use this command.

TERMIDERR
RESP2 values:

23 The named terminal cannot be found.

SET TRACEDEST

Change tracing options.



Conditions: INVREQ, IOERR, NOSPACE, NOSTG, NOTAUTH

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 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 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 the 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.

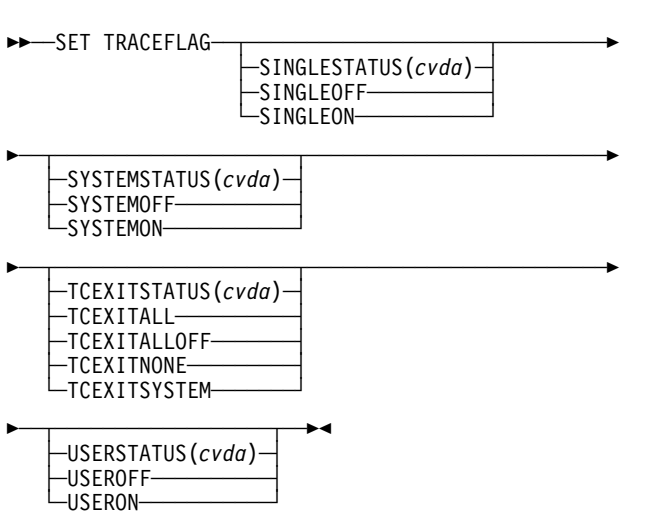
Examples

```
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.



Conditions: INVREQ, NOTAUTH

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 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.

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
Nonterminal-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:

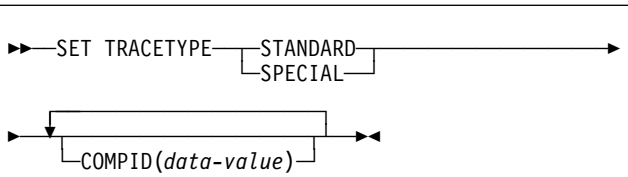
1 SYSTEMSTATUS has an invalid CVDA value.
2 USERSTATUS has an invalid CVDA value.
3 SINGLESTATUS has an invalid CVDA value.
4 TCEXITSTATUS has an invalid CVDA value.
5 TCEXITSTATUS is specified but VTAM is not installed.

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.



Conditions: NOTAUTH, NOTFND

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 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
BR	BRIDGE	3270 Bridge
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
KC		Task control
KE	KERNEL	Kernel
LC	LOCALCATLG	CICS local catalog manager
LD	LOADER	Program load manager
LG	LOGMGR	Log manager
LM	LOCKMGR	Lock manager
ME	MESSAGE	Message manager
MN	MONITOR	Monitoring manager
NQ		Enqueue domain
PA	PARAMGR	Parameter manager
PC		Program control
PG	PROGMGR	Program manager
RI		Resource manager interface (RMI)
RM		Recovery 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
WB	WEB	Web 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:

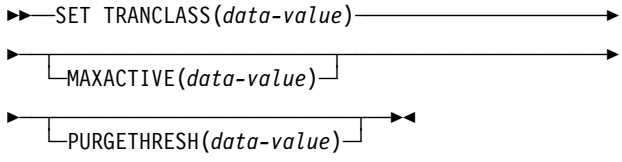
100 The user associated with the issuing task is not authorized to use this command.

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.



Conditions: INVREQ, NOTAUTH, TCIDERR

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 which 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, 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 DFHTCL*nn*, 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:

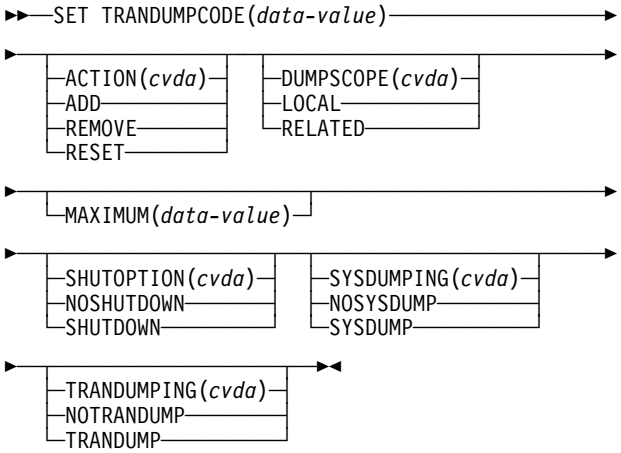
100 The user associated with the issuing task is not authorized to use this command.

TCIDERR
RESP2 values:

1 The transaction class cannot be found.

SET TRANDUMPCODE

Change an entry in the transaction dump table.



Conditions: DUPREC, INVREQ, IOERR, NOSPACE, NOTAUTH, NOTFND

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 the 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 an initial or 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 Problem Determination Guide*.

Options

ACTION(*cvda***)**
specifies the action 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 SYSDUMP, and only in a sysplex environment executing under MVS/ESA 5.1 and the MVS workload manager. In other systems, specifying RELATED causes 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.

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:

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.

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:

NOSYSDUMP
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, NOSYSDUMP 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.
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:

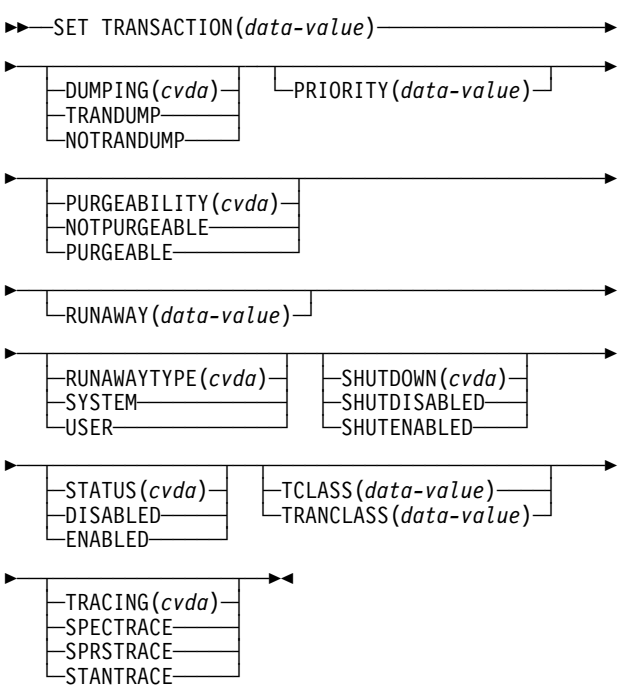
100 The user associated with the issuing task is not authorized to use this command.

NOTFND
RESP2 values:

1 The dump code cannot be found.

SET TRANSACTION

Change a TRANSACTION definition.



Conditions: INVREQ, NOTAUTH, TRANSIDERR

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 to abend 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 0–255.

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 tells you what the system value is.)

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 Transaction Server for OS/390, 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 DFHTCL*nn*, where *nn* is the number you specify, in the range 0 –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 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.

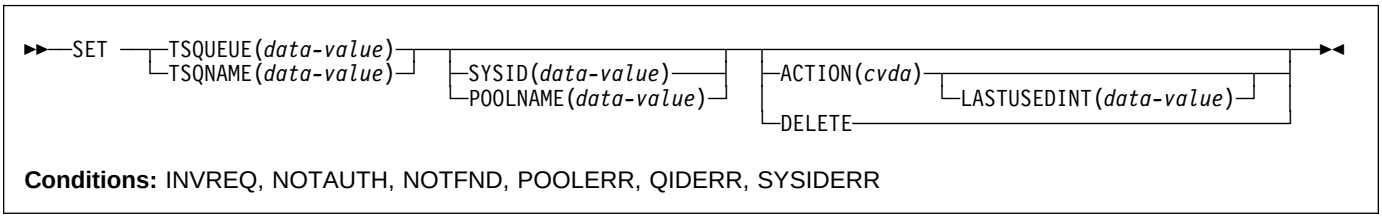
TRANSIDERR
RESP2 values:

1 The transaction cannot be found.

SET TSQUEUE / TSQNAME

Delete a TS queue.

This section applies also to the alternative command SET TSQNAME. Use either to delete a queue with a name up to 8 characters long, use SET TSQNAME to delete a queue with a name up to 16 characters long.



Conditions: INVREQ, NOTAUTH, NOTFND, POOLERR, QIDERR, SYSIDERR

Description

The SET TSQUEUE command enables you to delete a TS queue. The LASTUSEDINT option may be used to ensure that the queue to be deleted has not been referenced since a previous INQUIRE was issued. It may also be used to delete queues which have not been referenced within a given interval. If a queue is recoverable, a separate task must be attached to perform the deletion.

Options

ACTION(*cvda***)**
specifies the action to be taken on the queue. The CVDA value is:

DELETE
the queue is to be deleted.

LASTUSEDINT
if this option is specified, the queue is only deleted if its last used interval is greater or equal to the value specified.

POOLNAME(*data-value***)**
specifies an 8-character pool name.

SYSID(*data-value***)**
specifies a 4-character shared sysid.

TSQNAME(*data-value***)**
specifies the 1 to 16-character identifier of the TS queue.

TSQUEUE(*data-value***)**
specifies the 1 to 8-character identifier of the TS queue.

Conditions

INVREQ
RESP2 values:

1 The TSQUEUE was not deleted because LASTUSEDINT was greater than the interval, or because the the TSQUEUE is in use.
2 The action specified was not *DELETE*
3 LASTUSEDINT was specified but had an invalid value; that is, a negative value.
4 The filetype was not valid for the specified operation.

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 TSQUEUE cannot be found.

POOLERR
RESP2 values:

0 POOLNAME was specified but the pool could not be accessed.

QIDERR
RESP2 values:

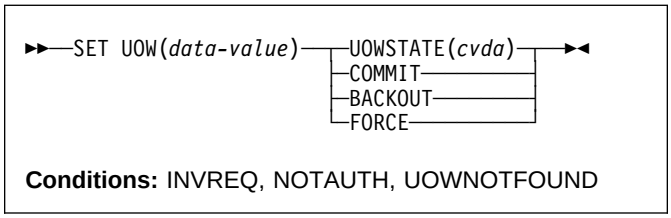
1 The QUEUE name was invalid; (it was binary zeros).

SYSIDERR
RESP2 values:

0 SYSID was specified but there is no corresponding pool, or the pool is unavailable.
3 The SYSID does not map to a shared pool.
4 Server error.
5 I/O error on coupling facility.

SET UOW

Commit, back out, or force a shunted unit of work.



Conditions: INVREQ, NOTAUTH, UOWNOTFOUND

Description

The SET UOW command enables you to commit, back out, or force a unit of work that has been shunted during the transaction's in-doubt period.

Options

UOW(*data-value***)**
specifies the 16-character identifier of the UOW to be committed, backed out, or forced.

UOWSTATE(*cvda***)**
specifies the action to be attempted for this UOW. CVDA values are:

BACKOUT
Attempt to force syncpoint backout processing, as specified for this UOW.

COMMIT
Attempt to force syncpoint commit processing, as specified for this UOW.

FORCE
Attempt to force the UOW to back out or commit, as specified on the ACTION option of the TRANSACTION resource definition.

Note: All these values are valid only for UOWs that are shunted in-doubt. For information about the INDOUBT attributes of TRANSACTION definitions, see the *CICS Resource Definition Guide*.

Conditions

INVREQ
RESP2 values:

3 UOWSTATE has an invalid CVDA value.
4 CICS is not in a valid state to COMMIT, BACKOUT, or FORCE this UOW.

NOTAUTH
RESP2 values:

100 The use of this command is not authorized.

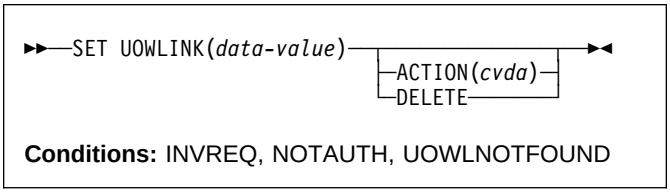
UOWNOTFOUND
RESP2 values:

1 The UOW cannot be found.

SET UOWLINK

Delete a link to a unit of work (a UOW-link) that was created by a connection that has since been discarded.

| UOWLINKs associated with RRS can be deleted when RRS
| has cold started.



Conditions: INVREQ, NOTAUTH, UOWLNOTFOUND

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

Description

The association between a unit of work and a connection is known as a UOW-link. You can use the INQUIRE UOWLINK command to browse all the UOW-links currently in the system. Some of the UOW-links may have been created by connections that have since been discarded. If so, you may be able to use the SET UOWLINK command to delete them. (For information about when it is safe to delete UOW-links, see the *CICS Intercommunication Guide*.)

Options

ACTION(*cvda***)**
specifies the action to be taken against the UOW-link. The CVDA value is:

DELETE
Delete the UOW-link.

UOWLINK(*data-value***)**
specifies the 4-character identifier of the UOW-connection dependency (the UOW-link) to which this command applies.

Conditions

INVREQ
RESP2 values:

2 Resynchronization is already in progress, or the UOW-link is already being processed by another instance of the SET UOWLINK command.

3 The unit of work is in-doubt, and the UOW-link is the coordinator of the commit or backout session. The unit of work must be forced using the SET UOW command before the UOW-link can be deleted.

4 This is not a link created by a connection, or is not a recoverable link.

5 The UOW-link (and the associated communication session) is still active.

6 ACTION has an invalid CVDA value.

7 The UOW-link has a suitable connection definition, and cannot be deleted.

NOTAUTH
RESP2 values:

100 The use of this command is not authorized.

UOWLNOTFOUND
RESP2 values:

1 The specified UOW-link cannot be found.

SET VOLUME

This command is supported in releases of CICS earlier than CICS Transaction Server for OS/390 to add, remove, or change availability of a named journal volume.

Description

SET VOLUME is obsolete, and is retained only for compatibility with previous releases. The only run-time support is to return the VOLIDERR condition. If this command is used, the translator translates it, but issues a warning message.

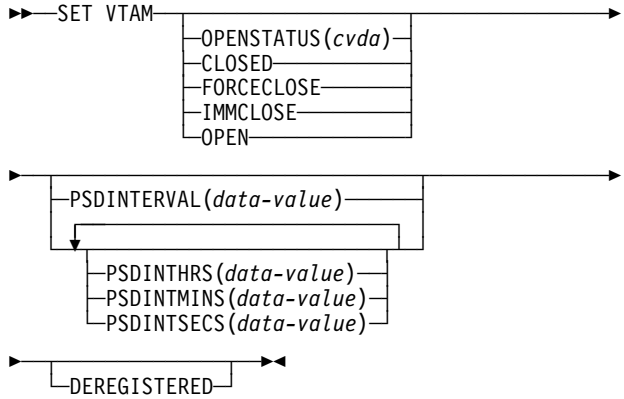
Conditions

VOLIDERR
RESP2 values:

1 The program has issued a SET VOLUME command. This command is withdrawn.

SET VTAM

Modify the CICS VTAM connection.



Conditions: INVREQ, IOERR, NOTAUTH

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
- Modify the persistent session delay interval value that CICS passes to VTAM
- Deregister CICS from membership of a VTAM generic resource.

Options

DEREGISTERED
specifies that CICS is to be removed from the VTAM generic resource that it is currently a member of. If you deregister a region from membership of a generic resource, you should end any affinities that it owns—see the PERFORM ENDAFFINITY command.

Generic resources are described in the *CICS Intercommunication Guide*.

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 shutdown 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 terminate abnormally as a result.

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.

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 System Definition Guide* for more about this option). CICS passes this value to VTAM whenever it opens the ACB. This occurs 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 *0hhmmss+*, 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 is 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.

16 Your attempt to deregister CICS from a VTAM generic resource failed because CICS is not registered as a member of a generic resource group.

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.

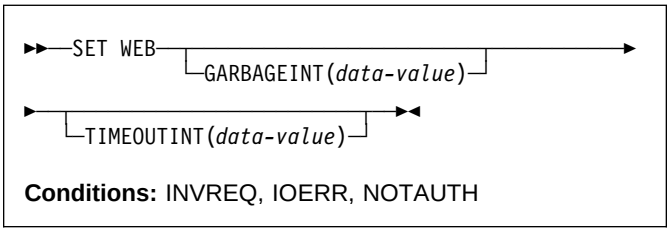
NOTAUTH

RESP2 values:

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

SET WEB

Modify CICS Web support.



Conditions: INVREQ, IOERR, NOTAUTH

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

Description

The SET WEB command allows you to:

- Change Web garbage collection settings.
- Change Web 3270 terminal timeout settings.

Options

GARBAGEINT(*data-value***)**
specifies, as a fullword, the interval in seconds at which the Web garbage collection task runs to clean up Web 3270 state data for which the terminal timeout interval has expired. The permitted range of values is 1 to 6000.

TIMEOUTINT(*data-value***)**
specifies, as a fullword, the period of time, in seconds, after which inactive Web 3270 sessions are eligible for garbage collection. The permitted range of values is 1 to 60.

Conditions

INVREQ
RESP2 values are:

11 an invalid value has been supplied for GARBAGEINT or TIMEOUTINT.

NOTAUTH
RESP2 values are:

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. CVDAs are described beginning on page 7.

This appendix consists of three tables:

- one, in alphabetic sequence of the CVDA values;
- one, (beginning on page 327) in numeric sequence of the CVDA values;
- and one, (beginning on page 332) gives the CVDA values returned by the INQUIRE TERMINAL|NETNAME DEVICE command.

CVDAs and numeric values in alphabetic sequence

CVDA	Value	CVDA	Value
ABEND	900	BASESPACE	664
ACQFAIL	515	BDAM	2
ACQUIRED	69	BEGINSESSION	510
ACQUIRING	71	BELOW	159
ACTIVE	181	BGAM	63
ACTIVITY	1002	BINARY	1038
ADD	291	BLK	47
ADDABLE	41	BLOCKED	16
ADDFAIL	519	BROWSABLE	39
ADVANCE	265	BSAM	61
ALARM	501	BTAM	62
ALLCONN	169	BUSY	612
ALLOCATED	81	C	149
ALLQUERY	431	CACHE	791
ALTERABLE	52	CANCEL	526
ALTERNATE	197	CANCELLED	624
ALTPRTCOPY	446	CANCELLING	1025
AND	1005	CD	491
ANY	158	CEDF	370
APLKYBD	391	CFTABLE	833
APLTEXT	393	CGROUP	902
APPC	124	CICS	660
APPCPARALLEL	374	CICSDATAKEY	379
APPCSINGLE	373	CICSEXECKEY	381
APPEND	1036	CICSTABLE	101
APPLICATION	559	CKOPEN	1055
ASACTL	224	CLEAR	640
ASCII7	616	CLIENTAUTH	1032
ASCII8	617	CLOSED	19
ASSEMBLER	150	CLOSELEAVE	261
ATI	75	CLOSEREQUEST	22
ATTENTION	524	CLOSING	21
AUDALARM	395	CMDPROT	673
AUTOACTIVE	630	CMDSECNO	205
AUTOARCH	262	CMDSECYES	207
AUTOCONN	170	COBOL	151
AUTOINACTIVE	631	COBOLII	375
AUTOPAGEABLE	80	COLD	788
AUTOSTART	618	COLDACQ	72
AUXILIARY	247	COLDQUERY	433
AUXPAUSE	313	COLDSTART	266
AUXSTART	312	COLOR	399
AUXSTOP	314	COMMIT	208
AVAILABLE	95	COMMITFAIL	792
BACKOUT	192	COMPLETE	1026
BACKTRANS	397	COMPOSITE	1003
BACKUPNONBWO	800	CONFFREE	82
BASE	10	CONFRECEIVE	83
BASEAPI	1052	CONFSEND	84

CVDA	Value
CONNECT	903
CONNECTED	690
CONNECTING	904
CONNECTION	755
CONSISTENT	723
CONSOLE	66
CONTENTION	836
CONTROLSHUT	623
CONVERSE	600
CONVIDLE	518
COORDINATOR	770
COPID	908
COPY	401
CREATE	67
CSIGN	905
CTERM	906
CTLGALL	632
CTLGMODIFY	633
CTLGNONE	634
CTRLABLE	56
CTX	907
CURRENT	260
CUSERID	909
DAE	684
DATA	508
DATASET	756
DATASETFULL	793
DATASTREAM	543
DEADLOCK	794
DEBUG	1082
DEC	46
DEFAULT	198
DEFRESP1	497
DEFRESP1OR2	528
DEFRESP2	498
DEFRESP3	499
DELAY	637
DELETABLE	43
DELETE	292
DELETEFAIL	520
DELEXITERROR	795
DEREGERROR	679
DEREGISTERED	678
DEST	235
DISABLED	24
DISABLING	25
DISCARDFAIL	513
DISCREQ	444
DISCONNING	910
DISK1	252
DISK2	253
DISK2PAUSE	254
DISPATCHABLE	228
DORMANT	1024
DPLSUBSET	383
DS3270	615
DUALCASE	403
DUMMY	781
DYNAMIC	178
EB	490
EBCDIC	1039
EMERGENCY	268
EMPTY	210
EMPTYREQ	31
ENABLED	23
ENDAFFINITY	790

CVDA	Value
EQUAL	911
ESDS	5
EVENT	334
EVENTFAIL	1012
EXCEPT	332
EXCEPTRESP	523
EXCI	650
EXCTL	48
EXECENQ	751
EXECENQADDR	752
EXIT	912
EXITTRACE	362
EXPIRED	1017
EXTENDEDDS	405
EXTRA	221
EXTSECURITY	194
FAILED	782
FAILEDBKOUT	357
FAILINGBKOUT	358
FCLOSE	273
FINALQUIESCE	183
FINPUT	270
FIRED	1001
FIRSTINIT	625
FIRSTQUIESCE	182
FIXED	12
FLUSH	783
FMH	502
FMHPARM	385
FOPEN	272
FORCE	342
FORCECANCEL	655
FORCECLOSE	351
FORCECLOSING	353
FORCED	1013
FORCEPURGE	237
FORMATEDF	606
FORMATTED	542
FORMFEED	407
FOUTPUT	271
FREE	85
FREEING	94
FULL	212
FULLAPI	384
FULLAUTO	1073
FWDRECOVABLE	354
GENERIC	651
GMT	604
GOINGOUT	172
GROUP	913
GTFSTART	317
GTFSTOP	318
HEURBACKOUT	758
HEURCOMMIT	757
HEX	45
HFORM	409
HIGH	914
HILIGHT	413
HOLD	163
HTTPNO	1034
HTTPYES	1033
IGNORE	1
IGNORERR	816
IMMCLOSE	350
IMMCLOSING	352
IMMQUIESCED	706

CVDA	Value
INACTIVE	378
INBOUND	547
INCOMPLETE	1014
INDEXRECFULL	796
INDIRECT	122
INDOUBT	620
INFLIGHT	621
INITCOMPLETE	628
INITIAL	789
INOUT	532
INPUT	226
INSERVICE	73
INSTALLED	550
INSTALLFAIL	512
INTERNAL	1058
INTRA	222
INTSTART	310
INTSTOP	311
INVALID	359
IOERROR	797
IRC	121
ISOLATE	658
JVM	1080
KATAKANA	415
KEYED	8
KSDS	6
LCKSTRUCFULL	832
LEAVE	811
LE370	377
LIC	493
LIGHTPEN	417
LOAD	834
LOCAL	605
LOCKING	837
LOG	54
LOGICAL	216
LOGTERM	269
LOSE	544
LOW	915
LPA	165
LUP	541
LUSTAT	525
LU61	125
MAIN	248
MAP	155
MAPSET	155
MCHCTL	241
MDT	506
MOD	813
MODEL	370
MORE	492
MSRCONTROL	419
MVS	780
NEGATIVE	530
NEWCOPY	167
NEWSESSION	485
NOALARM	500
NOALTPRTCOPY	447
NOAPLKYBD	392
NOAPLTEXT	394
NOAPPEND	1037
NOATI	76
NOAUDALARM	396
NOAUTO	1071
NOAUTOARCH	263
NOBACKTRANS	398

CVDA	Value
NOCEDF	371
NOCLEAR	641
NOCMDPROT	674
NOCOLOR	400
NOCONNECT	916
NOCONV	556
NOCONVERSE	601
NOCOPY	402
NOCREATE	68
NOCTL	223
NODAE	685
NODEBUG	1083
NODISCREQ	445
NODUALCASE	404
NOEMPTYREQ	32
NOEVENT	335
NOEXCEPT	333
NOEXCTL	49
NOEXIT	917
NOEXITTRACE	363
NOEXTENDEDDS	406
NOFMH	503
NOFMHPARM	386
NOFORCE	1054
NOFORMATEDF	607
NOFORMFEED	408
NOHFORM	410
NOHILIGHT	414
NOHOLD	164
NOISOLATE	657
NOJVM	1081
NOKATAKANA	416
NOLIGHTPEN	418
NOLOAD	835
NOLOG	55
NOLOSTLOCKS	710
NOMDT	507
NOMSGJRNL	531
NOMSRCONTROL	420
NONAUTOCONN	171
NONCICS	661
NONE	496
NONLE370	1084
NOOBFORMAT	422
NOOBOPERID	388
NOOUTLINE	424
NOPARTITIONS	426
NOPERF	331
NOPRESETSEC	243
NOPRINTADAPT	428
NOPROGSYMBOL	430
NOPRTCOPY	449
NOQUERY	432
NORECOVDATA	700
NOREENTPROT	681
NORELEASE	918
NORELREQ	443
NORETAINED	711
NORMAL	1016
NORMALBKOUT	356
NORMALRESP	522
NOROLLBACK	919
NOSECURITY	196
NOSHUTDOWN	289
NOSOSI	435
NOSPI	694

CVDA	Value
NOSSL	1031
NOSTSN	487
NOSWITCH	285
NOSYNCPOINT	603
NOSYSCONNECT	654
NOSYSDUMP	185
NOSYSLOG	784
NOTADDABLE	42
NOTALTERABLE	53
NOTAPPLIC	1
NOTASKSTART	608
NOTBROWSABLE	40
NOTBUSY	613
NOTCDEB	495
NOTCONNECTED	691
NOTCTRLABLE	57
NOTDEFINED	659
NOTDELETABLE	44
NOTDYNAMIC	1021
NOTEMPTY	211
NOTERMINAL	214
NOTEXTKYBD	437
NOTEXTPRINT	439
NOTFIRED	1000
NOTFWDRCVBLE	361
NOTINBOUND	546
NOTINIT	376
NOTINSTALLED	551
NOTKEYED	9
NOTLPA	166
NOTPENDING	127
NOTPURGEABLE	161
NOTRANDUMP	187
NOTREADABLE	36
NOTREADY	259
NOTRECOVABLE	30
NOTREQUIRED	667
NOTRLS	721
NOTROUTABLE	1022
NOTSOS	669
NOTSUPPORTED	15
NOTSUSPENDED	1027
NOTTABLE	100
NOTTI	78
NOTUPDATABLE	38
NOTWAIT	920
NOUCTRAN	451
NOUSER	1011
NOVALIDATION	441
NOVFORM	412
NOWAIT	341
NOWRITE	275
NOZCPTRACE	365
NRS	774
N906	931
N906D	921
OBFORMAT	421
OBOPERID	387
OBTAINING	96
OFF	200
OK	274
OLD	26
OLDCOPY	162
OLDSESSION	486
ON	201
OPEN	18

CVDA	Value
OPENAPI	1053
OPENERROR	798
OPENING	20
OPENOUTPUT	257
OPID	933
OR	1006
OUTLINE	423
OUTPUT	227
OUTSERVICE	74
OWNER	753
PAGEABLE	79
PARTITIONS	425
PARTITIONSET	156
PATH	11
PENDBEGIN	558
PENDDATA	560
PENDFREE	86
PENDING	126
PENDPASS	565
PENDRECEIVE	87
PENDRELEASE	562
PENDSTART	561
PENDSTSN	557
PENDUNSOL	564
PERF	330
PHASEIN	168
PHYSICAL	215
PLI	152
PL1	152
POOL	922
POSITIVE	529
POST	636
PRESETSEC	242
PRIMARY	110
PRINTADAPT	427
PRIVATE	174
PROCESS	1010
PROGAUTO	1072
PROGRAM	154
PROGSYMBOL	429
PROTECTED	504
PRTCOPY	448
PURGE	236
PURGEABLE	160
QUASIRENT	1050
QUEUE	814
QUIESCED	707
QUIESCING	708
QR	1057
READABLE	35
READBACK	209
READONLY	275
READY	258
REBUILD	923
RECEIVE	88
RECONNECT	924
RECOVDATA	701
RECOVERABLE	29
RECOVERED	277
RECOVERLOCKS	712
REENTPROT	680
REGERROR	677
REGISTERED	670
REJECT	815
RELATED	675
RELEASE	563

CVDA	Value
RELEASED	70
RELEASING	549
RELREQ	442
REMLOSTLOCKS	713
REMOTE	4
REMOVE	276
REMSESSION	740
REMTABLE	103
REPEATABLE	724
REQUIRED	666
REREAD	812
RESET	290
RESETLOCKS	714
RESSECINT	203
RESSECNO	202
RESSECYES	204
RESYNC	702
RETAINED	715
RETRY	716
REVERTED	264
REWIND	811
RLS	720
RLSACTIVE	730
RLSGONE	799
RLSINACTIVE	731
RLSSERVER	761
RMI	771
ROLLBACK	89
ROUTE	638
ROUTABLE	1023
RRCOMMITFAIL	830
RRDS	7
RRINDOUBT	831
RTR	527
RU	494
RUNNING	229
SCS	614
SECONDINIT	626
SEND	90
SESSION	372
SESSIONFAIL	517
SESSIONLOST	516
SETFAIL	514
SHARE	27
SHARED	173
SHUNTED	762
SHUTDISABLED	645
SHUTDOWN	288
SHUTENABLED	644
SIGN	925
SIGNEDOFF	245
SIGNEDON	244
SINGLEOFF	324
SINGLEON	323
SKIP	810
SMF	255
SOS	668
SOSABOVE	683
SOSBELOW	682
SOSI	434
SPECIFIC	652
SPECTRACE	177
SPI	693
SPRSTRACE	175
SQLCODE	926
SSL	1030
STANDBY	629
STANTRACE	176
START	635
STARTED	609
STARTING	772
STARTUP	180
STATIC	179
STOPPED	610
STSN	509
STSNSET	488
STSNTEST	489
SUBORDINATE	773
SUBSPACE	663
SURROGATE	371
SUSPENDED	231
SWITCH	188
SWITCHALL	287
SWITCHING	225
SWITCHNEXT	286
SYNCFREE	91
SYNCPOINT	602
SYNCRECEIVE	92
SYNCSEND	93
SYSCONNECT	653
SYSDUMP	184
SYSLOG	785
SYSTEM	643
SYSTEMOFF	320
SYSTEMON	319
TAKEOVER	111
TAPE1	250
TAPE2	251
TASK	233
TASKSTART	611
TCAM	64
TCAMSNA	65
TCEXITALL	366
TCEXITALLOFF	369
TCEXITNONE	368
TCEXITSYSTEM	367
TDQ	767
TERM	234
TERMINAL	213
TEXTKYBD	436
TEXTPRINT	438
THIRDINIT	627
THREADSAFE	1051
TIMEOUT	511
TIMER	1004
TPOOL	932
TPS55M3	553
TPS55M4	554
TPS55M5	555
TRANDUMP	186
TRANIDONLY	452
TSQ	768
TTI	77
TX	929
TXID	928
TWAIT	927
T3278M2	533
T3278M3	534
T3278M4	535
T3278M5	536
T3279M2	537
T3279M3	538

CVDA	Value
T3279M4	539
T3279M5	540
UCTRAN	450
UKOPEN	1056
UNATTEMPTED	820
UNAVAILABLE	672
UNBLOCKED	17
UNCOMMITTED	722
UNCONNECTED	703
UNDEFINED	14
UNDETERMINED	355
UNENABLED	33
UNENABLING	34
UNEXPECTED	1015
UNEXPIRED	1018
UNPROTECTED	505
UNQUIESCED	709
UNREGISTERED	671
UNSOLDATA	521
UOW	246
UPDATABLE	37
USER	642
USERDATAKEY	380
USEREXECKEY	382
USERID	930
USEROFF	322
USERON	321
USERTABLE	102
VALID	360
VALIDATION	440
VARIABLE	13
VFORM	411
VRRDS	732
VSAM	3
VTAM	60
WAIT	340
WAITCOMMIT	763
WAITER	754
WAITFORGET	622
WAITING	765
WAITRMI	766
WARMSTART	267
WIN	545
XCF	665
XM	123
XNOTDONE	144
XOK	143
ZCPTRACE	364

CVDAs and numeric values in numeric sequence

Value	CVDA
1	IGNORE
1	NOTAPPLIC
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
15	NOTSUPPORTED
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	NOTRECOVABLE
31	EMPTYREQ
32	NOEMPTYREQ
33	UNENABLED
34	UNENABLING
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
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
155	MAPSET
156	PARTITIONSET
158	ANY
159	BELOW
160	PURGEABLE
161	NOTPURGEABLE
162	OLDCOPY
163	HOLD
164	NOHOLD
165	LPA
166	NOTLPA
167	NEWCOPY

Value	CVDA
168	PHASEIN
169	ALLCONN
170	AUTOCONN
171	NONAUTOCONN
172	GOINGOUT
173	SHARED
174	PRIVATE
175	SPRSTRACE
176	STANTRACE
177	SPECTRACE
178	DYNAMIC
179	STATIC
180	STARTUP
181	ACTIVE
182	FIRSTQUIESCE
183	FINALQUIESCE
184	SYSDUMP
185	NOSYSDUMP
186	TRANDUMP
187	NOTRANDUMP
188	SWITCH
192	BACKOUT
194	EXTSECURITY
196	NOSECURITY
197	ALTERNATE
198	DEFAULT
200	OFF
201	ON
202	RESSECNO
203	RESSECINT
204	RESSECYES
205	CMDSECNO
207	CMDSECYES
208	COMMIT
209	READBACK
210	EMPTY
211	NOTEMPTY
212	FULL
213	TERMINAL
214	NOTERMINAL
215	PHYSICAL
216	LOGICAL
221	EXTRA
222	INTRA
223	NOCTL
224	ASACTL
225	SWITCHING
226	INPUT
227	OUTPUT
228	DISPATCHABLE
229	RUNNING
231	SUSPENDED
233	TASK
234	TERM
235	DEST
236	PURGE
237	FORCEPURGE
241	MCHCTL
242	PRESETSEC
243	NOPRESETSEC
244	SIGNEDON
245	SIGNEDOFF
246	UOW
247	AUXILIARY
248	MAIN
250	TAPE1
251	TAPE2
252	DISK1
253	DISK2
254	DISK2PAUSE
255	SMF
257	OPENOUTPUT
258	READY
259	NOTREADY
260	CURRENT
261	CLOSELEAVE
262	AUTOARCH
263	NOAUTOARCH
264	REVERTED
265	ADVANCE
266	COLDSTART
267	WARMSTART
268	EMERGENCY
269	LOGTERM
270	FINPUT
271	FOUTPUT
272	FOPEN
273	FCLOSE
274	OK
275	NOWRITE
275	READONLY
276	REMOVE
277	RECOVERED
285	NOSWITCH
286	SWITCHNEXT
287	SWITCHALL
288	SHUTDOWN
289	NOSHUTDOWN
290	RESET
291	ADD
292	DELETE
310	INTSTART
311	INTSTOP
312	AUXSTART
313	AUXPAUSE
314	AUXSTOP
317	GTFSTART
318	GTFSTOP
319	SYSTEMON
320	SYSTEMOFF
321	USERON
322	USEROFF
323	SINGLEON
324	SINGLEOFF
330	PERF
331	NOPERF
332	EXCEPT
333	NOEXCEPT
334	EVENT
335	NOEVENT
340	WAIT
341	NOWAIT
342	FORCE
350	IMMCLOSE
351	FORCECLOSE
352	IMMCLOSING
353	FORCECLOSING
354	FWDRECOVABLE
355	UNDETERMINED
356	NORMALBKOUT
357	FAILEDBKOUT
358	FAILINGBKOUT

Value	CVDA
359	INVALID
360	VALID
361	NOTFWDRCVBLE
362	EXITTRACE
363	NOEXITTRACE
364	ZCPTRACE
365	NOZCPTRACE
366	TCEXITALL
367	TCEXITSYSTEM
368	TCEXITNONE
369	TCEXITALLOFF
370	CEDF
370	MODEL
371	NOCEDF
371	SURROGATE
372	SESSION
373	APPCSINGLE
374	APPCPARALLEL
375	COBOLII
376	NOTINIT
377	LE370
378	INACTIVE
379	CICSDATAKEY
380	USERDATAKEY
381	CICSEXECKEY
382	USEREXECKEY
383	DPLSUBSET
384	FULLAPI
385	FMHPARM
386	NOFMHPARM
387	OBOPERID
388	NOOBOPERID
391	APLKYBD
392	NOAPLKYBD
393	APLTEXT
394	NOAPLTEXT
395	AUDALARM
396	NOAUDALARM
397	BACKTRANS
398	NOBACKTRANS
399	COLOR
400	NOCOLOR
401	COPY
402	NOCOPY
403	DUALCASE
404	NODUALCASE
405	EXTENDEDDS
406	NOEXTENDEDDS
407	FORMFEED
408	NOFORMFEED
409	HFORM
410	NOHFORM
411	VFORM
412	NOVFORM
413	HILIGHT
414	NOHILIGHT
415	KATAKANA
416	NOKATAKANA
417	LIGHTPEN
418	NOLIGHTPEN
419	MSRCONTROL
420	NOMSRCONTROL
421	OBFORMAT
422	NOOBFORMAT
423	OUTLINE
424	NOOUTLINE
425	PARTITIONS
426	NOPARTITIONS
427	PRINTADAPT
428	NOPRINTADAPT
429	PROGSYMBOL
430	NOPROGSYMBOL
431	ALLQUERY
432	NOQUERY
433	COLDQUERY
434	SOSI
435	NOSOSI
436	TEXTKYBD
437	NOTEXTKYBD
438	TEXTPRINT
439	NOTEXTPRINT
440	VALIDATION
441	NOVALIDATION
442	RELREQ
443	NORELREQ
444	DISCREQ
445	NODISCREQ
446	ALTPRTCOPY
447	NOALTPRTCOPY
448	PRTCOPY
449	NOPRTCOPY
450	UCTRAN
451	NOUCTRAN
452	TRANIDONLY
485	NEWSESSION
486	OLDSESSION
487	NOSTSN
488	STSNSET
489	STSNTEST
490	EB
491	CD
492	MORE
493	LIC
494	RU
495	NOTCDEB
496	NONE
497	DEFRESP1
498	DEFRESP2
499	DEFRESP3
500	NOALARM
501	ALARM
502	FMH
503	NOFMH
504	PROTECTED
505	UNPROTECTED
506	MDT
507	NOMDT
508	DATA
509	STSN
510	BEGINSESSION
511	TIMEOUT
512	INSTALLFAIL
513	DISCARDFAIL
514	SETFAIL
515	ACQFAIL
516	SESSIONLOST
517	SESSIONFAIL
518	CONVIDLE
519	ADDFAIL
520	DELETEFAIL
521	UNSOLDATA
522	NORMALRESP

Value	CVDA
523	EXCEPTRESP
524	ATTENTION
525	LUSTAT
526	CANCEL
527	RTR
528	DEFRESP1OR2
529	POSITIVE
530	NEGATIVE
531	NOMSGJRNL
532	INOUT
533	T3278M2
534	T3278M3
535	T3278M4
536	T3278M5
537	T3279M2
538	T3279M3
539	T3279M4
540	T3279M5
541	LUP
542	FORMATTED
543	DATASTREAM
544	LOSE
545	WIN
546	NOTINBOUND
547	INBOUND
549	RELEASING
550	INSTALLED
551	NOTINSTALLED
552	TPS55M2
553	TPS55M3
554	TPS55M4
555	TPS55M5
556	NOCONV
557	PENDSTSN
558	PENDBEGIN
559	APPLICATION
560	PENDDATA
561	PENDSTART
562	PENDRELEASE
563	RELEASE
564	PENDUNSOL
565	PENDPASS
600	CONVERSE
601	NOCONVERSE
602	SYNCPOINT
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
620	INDOUBT
621	INFLIGHT
622	WAITFORGET
623	CONTROLSHUT
624	CANCELLED
625	FIRSTINIT
626	SECONDINIT
627	THIRDINIT
628	INITCOMPLETE
629	STANDBY
630	AUTOACTIVE
631	AUTOINACTIVE
632	CTLGALL
633	CTLGMODIFY
634	CTLGNONE
635	START
636	POST
637	DELAY
638	ROUTE
640	CLEAR
641	NOCLEAR
642	USER
643	SYSTEM
644	SHUTENABLED
645	SHUTDISABLED
650	EXCI
651	GENERIC
652	SPECIFIC
653	SYSCONNECT
654	NOSYSCONNECT
655	FORCECANCEL
657	NOISOLATE
658	ISOLATE
659	NOTDEFINED
660	CICS
661	NONCICS
663	SUBSPACE
664	BASESPACE
665	XCF
666	REQUIRED
667	NOTREQUIRED
668	SOS
669	NOTSOS
670	REGISTERED
671	UNREGISTERED
672	UNAVAILABLE
673	CMDPROT
674	NOCMDPROT
675	RELATED
677	REGERROR
678	DEREGISTERED
679	DEREGERROR
680	REENTPROT
681	NOREENTPROT
682	SOSBELOW
683	SOSABOVE
684	DAE
685	NODAE
690	CONNECTED
691	NOTCONNECTED
693	SPI
694	NOSPI
700	NORECOVDATA
701	RECOVDATA
702	RESYNC
703	UNCONNECTED
706	IMMQUIESCED
707	QUIESCED
708	QUIESCING
709	UNQUIESCED
710	NOLOSTLOCKS

Value	CVDA
711	NORETAINED
712	RECOVERLOCKS
713	REMLOSTLOCKS
714	RESETLOCKS
715	RETAINED
716	RETRY
720	RLS
721	NOTRLS
722	UNCOMMITTED
723	CONSISTENT
724	REPEATABLE
730	RLSACTIVE
731	RLSINACTIVE
732	VRRDS
740	REMSESSION
751	EXECENQ
752	EXECENQADDR
753	OWNER
754	WAITER
755	CONNECTION
756	DATASET
757	HEURCOMMIT
758	HEURBACKOUT
761	RLSSERVER
762	SHUNTED
763	WAITCOMMIT
765	WAITING
766	WAITRMI
767	TDQ
768	TSQ
770	COORDINATOR
771	RMI
772	STARTING
773	SUBORDINATE
774	NRS
780	MVS
781	DUMMY
782	FAILED
783	FLUSH
784	NOSYSLOG
785	SYSLOG
788	COLD
789	INITIAL
790	ENDAFFINITY
791	CACHE
792	COMMITFAIL
793	DATASETFULL
794	DEADLOCK
795	DELEXITERROR
796	INDEXRECFULL
797	IOERROR
798	OPENERROR
799	RLSGONE
800	BACKUPNONBWO
810	SKIP
811	REWIND
811	LEAVE
812	REREAD
813	MOD
814	QUEUE
815	REJECT
816	IGNORERR
820	UNATTEMPTED
830	RRCOMMITFAIL
831	RRINDOUBT
832	LCKSTRUCFULL
833	CFTABLE
834	LOAD
835	NOLOAD
836	CONTENTION
837	LOCKING
900	ABEND
902	CGROUP
903	CONNECT
904	CONNECTING
905	CSIGN
906	CTERM
907	CTX
908	COPID
909	CUSERID
910	DISCONNING
911	EQUAL
912	EXIT
913	GROUP
914	HIGH
915	LOW
916	NOCONNECT
917	NOEXIT
918	NORELEASE
919	NOROLLBACK
920	NOTWAIT
921	N906D
922	POOL
923	REBUILD
924	RECONNECT
925	SIGN
926	SQLCODE
927	TWAIT
928	TXID
929	TX
930	USERID
931	N906
932	TPOOL
933	OPID
1000	NOTFIRED
1001	FIRED
1002	ACTIVITY
1003	COMPOSITE
1004	TIMER
1005	AND
1006	OR
1010	PROCESS
1011	NOUSER
1012	EVENTFAIL
1013	FORCED
1014	INCOMPLETE
1015	UNEXPECTED
1016	NORMAL
1017	EXPIRED
1018	UNEXPIRED
1021	NOTDYNAMIC
1022	NOTROUTABLE
1023	ROUTABLE
1024	DORMANT
1025	CANCELLING
1026	COMPLETE
1027	NOTSUSPENDED
1030	SSL
1031	NOSSL
1032	CLIENTAUTH
1033	HTTPYES
1034	HTTPNO

Value	CVDA
1036	APPEND
1037	NOAPPEND
1038	BINARY
1039	EBCDIC
1050	QUASIRENT
1051	THREADSAFE
1052	BASEAPI
1053	OPENAPI
1054	NOFORCE
1055	CKOPEN
1056	UKOPEN
1057	QR
1058	INTERNAL
1071	NOAUTO
1072	PROGAUTO
1073	FULLAUTO
1080	JVM
1081	NOJVM
1082	DEBUG
1083	NODEBUG
1084	NONLE370

CVDA values for the DEVICE option

CVDA sequence		CVDA sequence	
BATCHLU	191	T3286R	148
BIPROG	160	T3600BI	138
BISYNCH	128	T3601	177
CDRDLPRT	24	T3614	178
CONTNLU	189	T3650ATT	186
HARDCOPY	32	T3650PIPE	184
INTACTLU	190	T3650USER	187
ISCMMCONV	209	T3653HOST	185
LUCMODGRP	210	T3735	136
LUCSESS	211	T3740	137
LUTYPE4	193	T3780	133
LUTYPE6	192	T3790	180
MAGTAPE	20	T3790SCSP	182
RESSYS	208	T3790UP	181
SDLC	176	T7770	1
SEQDISK	18	VIDEOTERM	64
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		

Numeric sequence	
1	T7770
2	SYSTEM7
8	TCONSOLE
18	SEQDISK
20	MAGTAPE
24	CDRDLPRT
32	HARDCOPY
33	TWX3335
34	TELETYPE
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	ISCMMCONV
210	LUCMODGRP
211	LUCSESS

Appendix B. EXEC interface block (EIB) response and function codes

This appendix lists the response codes and the function codes of EXEC CICS commands.

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 (Byte 3)	RESP Value code code	EIBRCODE	Abend
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
UOWNOTFOUND	102	66	-
USERIDERR	69	45	AEYX
VOLIDERR	71	47	AEXV

EXEC CICS DISABLE, ENABLE, and EXTRACT EXIT commands

Conditions that can be raised by the DISABLE, ENABLE, and EXTRACT EXIT commands are INVEXITREQ and NOTAUTH. There are no conditions associated with the RESYNC command.

Condition	RESP Value	EIBRCODE	Abend
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 in command sequence, then in function code sequence. For information about other function codes, see the *CICS Application Programming Reference* manual.

Command	Code
ACQUIRE TERMINAL	86 02
COLLECT STATISTICS	70 08
CREATE CONNECTION	30 0E
CREATE DB2CONN	30 20
CREATE DB2ENTRY	30 22
CREATE DB2TRAN	30 24
CREATE DOCTEMPLATE	30 2E
CREATE ENQMODEL	30 2A
CREATE FILE	30 14
CREATE JOURNALMODEL	30 1E
CREATE LSRPOOL	30 16
CREATE MAPSET	30 04
CREATE PARTITIONSET	30 06
CREATE PARTNER	30 18
CREATE PROCESSTYPE	30 26
CREATE PROFILE	30 0A
CREATE PROGRAM	30 02
CREATE REQUESTMODEL	30 2C
CREATE SESSIONS	30 12
CREATE TCPIPSERVICE	30 30
CREATE TDQUEUE	30 1C
CREATE TERMINAL	30 10
CREATE TRANCLASS	30 1A
CREATE TRANSACTION	30 08
CREATE TSMODEL	30 28
CREATE TYPETERM	30 0C
DISABLE PROGRAM	22 04
DISCARD AUTINSTMODEL	42 10
DISCARD CONNECTION	58 10
DISCARD DB2CONN	94 10
DISCARD DB2ENTRY	94 30
DISCARD DB2TRAN	94 50
DISCARD DOCTEMPLATE	9E 10
DISCARD ENQMODEL	90 90
DISCARD FILE	4C 10
DISCARD JOURNALMODEL	92 10
DISCARD JOURNALNAME	60 10
DISCARD PARTNER	44 10
DISCARD PROCESSTYPE	96 10
DISCARD PROFILE	46 10
DISCARD PROGRAM	4E 10
DISCARD REQUESTMODEL	9A 10
DISCARD TCPIPSERVICE	9C 10
DISCARD TDQUEUE	5C 10
DISCARD TERMINAL	52 10
DISCARD TRANCLASS	5E 18
DISCARD TRANSACTION	50 10
DISCARD TSMODEL	80 30
ENABLE PROGRAM	22 02
EXTRACT EXIT	22 06

Command	Code
INQUIRE ACTIVITYID	96 12
INQUIRE AUTINSTMODEL	42 02
INQUIRE AUTOINSTALL	68 12
INQUIRE CFDTPOOL	98 02
INQUIRE CONNECTION	58 02
INQUIRE CONTAINER	96 14
INQUIRE DB2CONN	94 02
INQUIRE DB2ENTRY	94 22
INQUIRE DB2TRAN	94 42
INQUIRE DELETSHIPPED	68 22
INQUIRE DOCTEMPLATE	9E 02
INQUIRE DSNAME	7A 02
INQUIRE DUMPDS	66 02
INQUIRE ENQMODEL	90 82
INQUIRE EVENT	96 16
INQUIRE EXCI	7C 02
INQUIRE EXITPROGRAM	88 02
INQUIRE FILE	4C 02
INQUIRE IRC	6E 02
INQUIRE JOURNALMODEL	92 02
INQUIRE JOURNALNAME	60 12
INQUIRE JOURNALNUM	60 02
INQUIRE MODENAME	5A 02
INQUIRE MONITOR	70 12
INQUIRE NETNAME	52 16
INQUIRE NETNAME	52 06
INQUIRE PARTNER	44 02
INQUIRE PROCESS	96 18
INQUIRE PROCESSTYPE	96 02
INQUIRE PROFILE	46 02
INQUIRE PROGRAM	4E 02
INQUIRE REQID	8A 02
INQUIRE REQUESTMODEL	9A 02
INQUIRE RRMS	3A 02
INQUIRE STATISTICS	70 02
INQUIRE STORAGE	5E 08
INQUIRE STREAMNAME	92 12
INQUIRE SYSDUMPCODE	66 22
INQUIRE SYSTEM	54 02
INQUIRE TASK	5E 02
INQUIRE TCLASS	5E 12
INQUIRE TCPIP	9C 12
INQUIRE TCPIPSERVICE	9C 02
INQUIRE TDQUEUE	5C 02
INQUIRE TERMINAL	52 02
INQUIRE TERMINAL	52 12
INQUIRE TIMER	96 38
INQUIRE TRACEDEST	78 02
INQUIRE TRACEFLAG	78 12
INQUIRE TRACETYPE	78 22
INQUIRE TRANCLASS	5E 1A

Command	Code
INQUIRE TRANDUMPCODE	66 12
INQUIRE TRANSACTION	50 02
INQUIRE TSMODEL	80 22
INQUIRE TSPOOL	80 1A
INQUIRE TSQNAME	80 12
INQUIRE TSQUEUE	80 02
INQUIRE UOW	90 02
INQUIRE UOWDSNFAIL	90 62
INQUIRE UOWENQ	90 22
INQUIRE UOWLINK	90 42
INQUIRE VOLUME	62 02
INQUIRE VTAM	68 02
INQUIRE WEB	9C 22
PERFORM DELETSHIPPED	68 26
PERFORM DUMP	7E 04
PERFORM ENDAFFINITY	58 06
PERFORM RESETTIME	72 02
PERFORM SECURITY	64 02
PERFORM SHUTDOWN	76 02
PERFORM STATISTICS	70 06
RESYNC ENTRYNAME	16 04
SET AUTOINSTALL	68 14
SET CONNECTION	58 04
SET DB2CONN	94 04
SET DB2ENTRY	94 24
SET DB2TRAN	94 44
SET DELETSHIPPED	68 24
SET DSNAME	7A 04
SET DUMPDS	66 04
SET ENQMODEL	90 84
SET FILE	4C 04
SET IRC	6E 04
SET JOURNALNAME	60 14
SET JOURNALNUM	60 04
SET MODENAME	5A 04
SET MONITOR	70 14
SET NETNAME	52 08
SET PROCESSTYPE	96 04
SET PROGRAM	4E 04
SET STATISTICS	70 04
SET SYSDUMPCODE	66 24
SET SYSTEM	54 04
SET TASK	5E 04
SET TCLASS	5E 14
SET TCPIP	9C 14
SET TCPIPSERVICE	9C 04
SET TDQUEUE	5C 04
SET TERMINAL	52 04
SET TERMINAL	52 14
SET TRACEDEST	78 04
SET TRACEFLAG	78 14
SET TRACETYPE	78 24
SET TRANCLASS	5E 1C
SET TRANDUMPCODE	66 14
SET TRANSACTION	50 04
SET TSQNAME	80 14
SET TSQUEUE	80 04
SET UOW	90 04
SET UOWLINK	90 44
SET VOLUME	62 04
SET VTAM	68 04
SET WEB	9C 24

Code	Command
16 04	RESYNC ENTRYNAME
22 02	ENABLE PROGRAM
22 04	DISABLE PROGRAM
22 06	EXTRACT EXIT
30 0A	CREATE PROFILE
30 0C	CREATE TYPETERM
30 0E	CREATE CONNECTION
30 02	CREATE PROGRAM
30 04	CREATE MAPSET
30 06	CREATE PARTITIONSET
30 08	CREATE TRANSACTION
30 1A	CREATE TRANCLASS
30 1C	CREATE TDQUEUE
30 1E	CREATE JOURNALMODEL
30 10	CREATE TERMINAL
30 12	CREATE SESSIONS
30 14	CREATE FILE
30 16	CREATE LSRPOOL
30 18	CREATE PARTNER
30 20	CREATE DB2CONN
30 22	CREATE DB2ENTRY
30 24	CREATE DB2TRAN
30 26	CREATE PROCESSTYPE
30 28	CREATE TSMODEL
30 2A	CREATE ENQMODEL
30 2C	CREATE REQUESTMODEL
30 2E	CREATE DOCTEMPLATE
30 30	CREATE TCPIPSERVICE
3A 02	INQUIRE RRMS
42 02	INQUIRE AUTINSTMODEL
42 10	DISCARD AUTINSTMODEL
44 02	INQUIRE PARTNER
44 10	DISCARD PARTNER
46 02	INQUIRE PROFILE
46 10	DISCARD PROFILE
4C 02	INQUIRE FILE
4C 04	SET FILE
4C 10	DISCARD FILE
4E 02	INQUIRE PROGRAM
4E 04	SET PROGRAM
4E 10	DISCARD PROGRAM
50 02	INQUIRE TRANSACTION
50 04	SET TRANSACTION
50 10	DISCARD TRANSACTION
52 02	INQUIRE TERMINAL
52 04	SET TERMINAL
52 06	INQUIRE NETNAME
52 08	SET NETNAME
52 10	DISCARD TERMINAL
52 12	INQUIRE TERMINAL
52 14	SET TERMINAL
52 16	INQUIRE NETNAME
54 02	INQUIRE SYSTEM
54 04	SET SYSTEM
58 02	INQUIRE CONNECTION
58 04	SET CONNECTION
58 06	PERFORM ENDAFFINITY
58 10	DISCARD CONNECTION
5A 02	INQUIRE MODENAME
5A 04	SET MODENAME
5C 02	INQUIRE TDQUEUE
5C 04	SET TDQUEUE
5C 10	DISCARD TDQUEUE
5E 02	INQUIRE TASK
5E 04	SET TASK
5E 08	INQUIRE STORAGE
5E 1A	INQUIRE TRANCLASS
5E 1C	SET TRANCLASS
5E 12	INQUIRE TCLASS
5E 14	SET TCLASS
5E 18	DISCARD TRANCLASS
60 02	INQUIRE JOURNALNUM
60 04	SET JOURNALNUM
60 10	DISCARD JOURNALNAME
60 12	INQUIRE JOURNALNAME
60 14	SET JOURNALNAME
62 02	INQUIRE VOLUME
62 04	SET VOLUME
64 02	PERFORM SECURITY
66 02	INQUIRE DUMPDS
66 04	SET DUMPDS
66 12	INQUIRE TRANDUMPCODE
66 14	SET TRANDUMPCODE
66 22	INQUIRE SYSDUMPCODE
66 24	SET SYSDUMPCODE
68 02	INQUIRE VTAM
68 04	SET VTAM
68 12	INQUIRE AUTOINSTALL
68 14	SET AUTOINSTALL
68 22	INQUIRE DELETSHIPPED
68 24	SET DELETSHIPPED
68 26	PERFORM DELETSHIPPED
6E 02	INQUIRE IRC
6E 04	SET IRC
70 02	INQUIRE STATISTICS
70 04	SET STATISTICS
70 06	PERFORM STATISTICS
70 08	COLLECT STATISTICS
70 12	INQUIRE MONITOR
70 14	SET MONITOR
72 02	PERFORM RESETTIME
76 02	PERFORM SHUTDOWN
78 02	INQUIRE TRACEDEST
78 04	SET TRACEDEST
78 12	INQUIRE TRACEFLAG
78 14	SET TRACEFLAG
78 22	INQUIRE TRACETYPE
78 24	SET TRACETYPE
7A 02	INQUIRE DSNAME
7A 04	SET DSNAME
7C 02	INQUIRE EXCI
7E 04	PERFORM DUMP
80 02	INQUIRE TSQUEUE
80 04	SET TSQUEUE
80 12	INQUIRE TSQNAME
80 14	SET TSQNAME
80 1A	INQUIRE TSPOOL
80 22	INQUIRE TSMODEL
80 30	DISCARD TSMODEL
86 02	ACQUIRE TERMINAL
88 02	INQUIRE EXITPROGRAM
8A 02	INQUIRE REQID
90 02	INQUIRE UOW
90 04	SET UOW
90 22	INQUIRE UOWENQ
90 42	INQUIRE UOWLINK
90 44	SET UOWLINK
90 62	INQUIRE UOWDSNFAIL
90 82	INQUIRE ENQMODEL
90 84	SET ENQMODEL
90 90	DISCARD ENQMODEL
92 02	INQUIRE JOURNALMODEL

Code	Command
92 10	DISCARD JOURNALMODEL
92 12	INQUIRE STREAMNAME
94 02	INQUIRE DB2CONN
94 04	SET DB2CONN
94 10	DISCARD DB2CONN
94 22	INQUIRE DB2ENTRY
94 24	SET DB2ENTRY
94 30	DISCARD DB2ENTRY
94 42	INQUIRE DB2TRAN
94 44	SET DB2TRAN
94 50	DISCARD DB2TRAN
96 02	INQUIRE PROCESSTYPE
96 04	SET PROCESSTYPE
96 10	DISCARD PROCESSTYPE
96 12	INQUIRE ACTIVITYID
96 14	INQUIRE CONTAINER
96 16	INQUIRE EVENT
96 18	INQUIRE PROCESS
96 38	INQUIRE TIMER
98 02	INQUIRE CFDTPOOL
9A 02	INQUIRE REQUESTMODEL
9A 10	DISCARD REQUESTMODEL
9C 02	INQUIRE TCPIPSERVICE
9C 04	SET TCPIPSERVICE
9C 10	DISCARD TCPIPSERVICE
9C 12	INQUIRE TCPIP
9C 14	SET TCPIP
9C 22	INQUIRE WEB
9C 24	SET WEB
9E 02	INQUIRE DOCTEMPLATE
9E 10	DISCARD DOCTEMPLATE

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 6 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 Table 7 on page 346, Table 8 on page 346, Table 9 on page 348, Table 10 on page 349, and Table 11 on page 350.

Table 6 (Page 1 of 5). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
		Codes caused by syntactical errors
n,400	DFHCA5211	A misplaced delimiter occurs in ATTRIBUTES. The invalid delimiter is at offset *n* in the ATTRIBUTES string.
n,401	DFHCA5204	A keyword specified within ATTRIBUTES is invalid. The invalid keyword is at offset *n* in the ATTRIBUTES string.
n,402	DFHCA5212, DFHCA5213	A keyword within ATTRIBUTES cannot be uniquely identified from its abbreviation. The invalid keyword is at offset *n* in the ATTRIBUTES string.
k,403	DFHCA5501	A required keyword is omitted. The omitted keyword has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
404	DFHCA5529	A required keyword is omitted. The omitted keyword must be selected from two mutually exclusive keywords, as specified in the associated message.
k,405	DFHCA5504	One specified keyword requires another one to be specified. The omitted keyword has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,406	DFHCA5206	A keyword occurs more than once within ATTRIBUTES. The duplicate keyword has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,407	DFHCA5503 DFHCA5506	Conflicting keywords are specified. The keyword causing the conflict has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,410	DFHCA5210 DFHCA5519 DFHCA5521 DFHCA5522 DFHCA5526 DFHCA5528 DFHCA5532	An invalid operand is supplied for a keyword within ATTRIBUTES. The keyword in error has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,410	DFHCA5542	Length of Remoteprefix and length of Prefix must be the same The keyword in error has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,410	DFHCA5543	Generics must be in the same place in the Prefix and in the Remoteprefix. The keyword in error has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,411	DFHCA5207	An operand is supplied for a keyword that does not need one. The keyword in error has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,412	DFHCA5205	A required operand for a keyword within ATTRIBUTES is omitted. The keyword in error has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,413	DFHCA5517	The operands of two or more keywords conflict with one another. The first conflicting keyword detected has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,414	DFHCA5507	The value of the operand of a keyword within ATTRIBUTES is too small. The keyword in error has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.

Table 6 (Page 2 of 5). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
k,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 *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
k,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 *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
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.
k,419	DFHCA5217	A closing parenthesis has been omitted from a DESCRIPTION keyword within ATTRIBUTES. The keyword in error (DESCRIPTION) has code *k* in Table 7, Table 8, Table 9, Table 10, or Table 11.
420	DFHCA5508	PROTECTNUM must be less than or equal to THREADLIMIT, or COMTHREADLIM must be less than or equal to TCBLIMIT, or THREADLIMIT must be less than or equal to TCBLIMIT.
		Codes caused by errors deleting existing resources
500	DFHAM4803 DFHAM4834 DFHAM4836 DFHAM4842 DFHAM4896 DFHZC5913	Install failed because the resource is currently in use.
500	DFHAM4834 DFHAM4838	Install failed because not disabled.
500	DFHAM4853	Another DB2TRAN is installed with the same transid, or failed to install implicit DB2TRAN because there is an existing DB2TRAN installed with the same transid.
500	DFHAM4874	install failed because the attribute already exists.
500	DFHAM4894	install failed because the resource is not disabled.
500	DFHAM4903	install failed because the service is open.
501	DFHAM4841	Install failed because definition of *restype resname* is in use by task no. *taskno* (transaction id. *tranid*).
501	DFHZC5980	Resource *resource* is in use by task *taskid* Transaction *tranid*.
502	DFHZC6304	Deletion of remote terminal *termid* failed because it is in use by another transaction.
503	DFHZC5915	Deletion of *restype id* 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 *termid* failed. It has pending DFHZCP activity.
505	DFHZC5918	Deletion of terminal *termid* Console *consname* failed. It has pending DFHZCP activity.
506	DFHZC5914	Deletion of terminal *termid* found another deletion of it in progress.
506	DFHZC5937	Deletion of modename *modename* found another deletion of it in progress.
507	DFHZC5902	Deletion of terminal *termid* failed. BMS Paging session still active.
508	DFHZC5917	Deletion of terminal *termid* failed. Error message writer still active.
509	DFHZC5904	Deletion of terminal *termid* failed. CEDF is still active.
510	DFHZC5941	Install for terminal *termid* failed. Console *consname* has a conversation outstanding.
511	DFHZC5907	Deletion of remote shipped terminal failed for connection *cccc*.
512	DFHZC5925	Deletion of connection *cccc* failed. Its AID-Chains are not empty.
513	DFHZC5929	Deletion of connection *cccc* failed. It is in use by *n* indirect connections.
514	DFHZC5938	Deletion of modename *modename* failed. Unable to delete sessions.
515	DFHZC5951	Deletion of connection *ssss* failed. Unable to delete sessions.
516	DFHZC5945	Deletion of sessions *ssss* failed. Connection *cccc* is defined to IRC.
517	DFHZC5952	Deletion of terminal *termid* failed. It needs to be SET RELEASED.
518	DFHZC5969	Deletion of dependent modenames failed for connection *modename*.
519	DFHZC5974	Deletion of pool *pppp* failed. Unable to delete pool entries.
520	DFHZC5979	Deletion of pool *pppp* failed. It still has session *termid*.
520	DFHZC5982	Deletion of pool *pppp* failed. Pool entry is in use for *termid*.

Table 6 (Page 3 of 5). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
521	DFHZC5958	Install failed for *xxxx*. This is the name of the local system, which must not be replaced.
522	DFHZC5940	Install for terminal *termid* failed. Error console cannot be deleted.
523	DFHZC5989	Deletion of resource *resource* failed. Remote deletion in connection *cccc* failed.
524	DFHZC5943	MRO connection *conname* cannot be deleted because IRC is open.
		Codes caused by errors in installing the new resource
600	DFHTO6000	The definition for TERMINAL *termdef* refers to an undefined TYPETERM *termtype*.
600	DFHTO6001	The definition for pooled TERMINAL *termdef* refers to an undefined TYPETERM *termtype*.
601	DFHAM4910	The install failed because the member was not found in the partitioned data set.
601	DFHTO6002	The definition for SESSIONs *sesdef* refers to an undefined CONNECTION *condef*.
601	DFHZC5911	Install for resource *resource* failed. Connection *cccc* not found.
601	DFHZC5932	Install for modename *modename* failed. Connection *cccc* not found.
602	DFHZC5962	Install for resource *resource* failed. Modename parameter not found.
603	DFHZC5906	Install failed because *xxxx* is not a permitted value for a terminal or connection name.
604	DFHZC5933	Install for modename *modename* failed. Connection *cccc* is not valid here.
605	DFHAM4889	Install of resource failed because an *attribute* is invalid.
606	DFHAM4890	Install of TDQUEUE *tdqname* failed because the TYPE has not been specified.
607	DFHAM4870	Install failed for program *progname* - language RPG is not supported under MVS.
608	DFHAM4832	Unable to open TDQUEUE *tdqname* because the DFHINTRA data set is not open.
608	DFHAM4909	The install failed because the DDNAME was not found.
609	DFHAM4904	The install failed because the port was in use.
609	DFHAM4908	The install failed because the templatename already exists.
610	DFHAM4905	The install failed because the option is not available in this system.
611	DFHAM4901	The install of a requestmodel failed because a duplicate pattern already exists.
620	DFHZC5912	Install for terminal *termid* failed. It is incompatible with connection *cccc*.
620	DFHZC5949	Install for sessions *ssss* failed. It is incompatible with connection *cccc*.
621	DFHZC5900	System *sysid* has shipped definitions but connection *cccc* is not known to this system.
622	DFHZC5921	Install of terminal *termid* failed. VTAM support not loaded.
622	DFHZC5988	Install for resource *resource* failed. VTAM support not generated.
623	DFHZC5909	Install of resource *resource* failed. Call to DFHIRP *irp_function Return_code* did not succeed, See DFHIRSDS for return code.
624	DFHZC5931	Install for modename *modename* failed. Maximum number of APPC sessions would have been exceeded.
625	DFHZC5973	Install for sessions *ssss* failed. Max session-count reached for modename *modename*.
626	DFHZC5955	SESNUMB greater than DLTHRED in the SIT (*nnnn*).
627	DFHZC5934	Install for modename *modename* failed. Single-session connection *cccc* is already in use.
628	DFHZC5936	Install for modename *modename* failed. Connection *cccc* has active modegroup *xxxx*.
629	DFHZC5939	Install for *name* failed. Duplicate session- or modegroup-name for connection *sysid*.
630	DFHZC5946	Install for sessions *ssss* failed. Connection *cccc* is defined to IRC.
631	DFHZC5948	Install for sessions *ssss* failed. Connection *cccc* is not suitable for IRC.
632	DFHZC5954	Install for resource *resource* failed. Unable to install sessions component.
633	DFHZC5963	*operation* RUSIZE *xxxx* from terminal *termid* was greater than TYPETERM RUSIZE *yyyy*.
634	DFHZC5967	Install for modename *modename* failed. Unable to install sessions.
635	DFHZC5968	Unable to install LU Services Manager for modename *modename*.
636	DFHZC5981	Pool *pppp* not found.

Table 6 (Page 4 of 5). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
637	DFHZC5985	Install for resource *resource* failed. Unable to install connection component.
638	DFHTO6003	TERMINAL *termdef* specifies CONSOLE but refers to TYPETERM *termtype* which does not specify DEVICE=CONSOLE.
639	DFHTO6004	TERMINAL *termdef* does not specify CONSOLE but refers to TYPETERM *termtype* which specifies DEVICE=CONSOLE.
640	DFHTO6005	PRINTER or ALTPRINTER for TERMINAL *termdef* is invalid for the DEVICE specified in TYPETERM *termtype*.
641	DFHTO6006	PRINTERCOPY or ALTPRINTERCOPY for TERMINAL *termdef* is invalid for the DEVICE specified in TYPETERM *termtype*.
642	DFHTO6007	AUTINSTMODEL YES|ONLY for TERMINAL *termdef* is invalid for the DEVICE specified in TYPETERM *termtype*.
643	DFHTO6008	
644	DFHTO6009	The definition for SESSIONs *sesdef* refers to CONNECTION *condef* which specifies a different PROTOCOL.
645	DFHTO6010	The definition for SESSIONs *sesdef* must specify PROTOCOL LU61 as it refers to an MRO CONNECTION *condef*.
646	DFHTO6011	SESSIONs *sesdef* must specify both SENDCOUNT and RECEIVECOUNT as it refers to an MRO CONNECTION *condef*.
647	DFHTO6013	No SESSIONs definition refers to CONNECTION *condef*.
648	DFHTO6014	POOL is required for TERMINAL *termdef* as it refers to TYPETERM *typedef* which specifies SESSIONTYPE=PIPELINE.
649	DFHTO6015	TRANSACTION for TERMINAL *termdef* is invalid for the DEVICE specified in TYPETERM *typedef*.
650	DFHTO6016	The MRO CONNECTION *condef* is referenced by more than one SESSIONs definition, including *sesdef*.
651	DFHTO6017	REMOTESYSTEM for TERMINAL *termid* is invalid for the DEVICE specified in TYPETERM *typeterm*.
652	DFHTO6018	TERMINAL *termid* refers to TYPETERM *typeterm* which has an invalid ALTSCREEN.
653	DFHTO6020	SESSIONS *sesdef* refers to single-session CONNECTION *condef* 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 *sesdef* specifies a send or receive count with no prefix.
656	DFHZC6301	Install for *tttt* failed. Duplicate netname *netname* for resource *rrrr* found.
657	DFHZC6302	Install for connection *cccc* failed. Duplicate netname *netname* for resource *rrrr* found.
658	DFHZC6303	Install for *tttt* failed. Duplicate netname *netname* found.
660	DFHZC6331	Install for connection *tttt* failed. Non-VTAM terminal with same name already exists.
660	DFHZC6332	Install for terminal *tttt* failed. Non-VTAM terminal with same name already exists.
661	DFHZC5950	Install for terminal *termid* failed. Console *consname* already exists.
662	DFHZC6310	Install for terminal *termid* failed. Console *consname* must be defined by ID not name.
663	DFHZC6311	Install for terminal *termid* failed. Console ID *conslid* does not map to a console name known to MVS.
664	DFHZC6330	Install for *tttt* failed. LDCLIST parameter *ldclist* not found.
665	DFHZC6333	INSTALL for modename *modename* failed. Zero sessions specified.
666	DFHAM4833 DFHZC6361	Resource cannot be installed with specified userID because of a security error.
667	DFHZC6362	Install for terminal *portname* with userID *userid* failed because the preset userID has been revoked.
668	DFHZC6363	Install for terminal *portname* with userID *userid* failed because the preset userID's group access has been revoked.
669	DFHZC6364	Install for terminal *portname* with userID *userid* failed because the ESM returned an unrecognized response.
670	DFHZC6365	Install for terminal *portname* with userID *userid* failed because the external security manager is inactive.
671	DFHZC6366	Install for terminal *portname* with userID *userid* failed because the userID is not authorized to access this CICS system.

Table 6 (Page 5 of 5). RESP2 values corresponding to messages

RESP2	Msgid	Description or message
672	DFHZC6367	Install for terminal *termid* with userID *userid* failed because the SECLABEL check failed.
673	DFHZC6368	Install for terminal *portname* with userID *userid* failed because the external security manager is quiesced.
674	DFHZC6369	Install for terminal *portname* failed because national language *langcode* is invalid.
675	DFHZC6370	Install for terminal *portname* failed because national language *langcode* is unavailable.
676	DFHZC6371	Install for terminal *portname* with userID *userid* failed because the userID is not authorized to use this portname.
677	DFHZC5944	Install for *type*(*id*) has failed. It would make a loop of connection definitions.
679	DFHAM4837	Install of DB2ENTRY or DB2TRAN failed because DB2CONN not installed.
680	DFHAM4850	DB2TRAN not installed because refers to a DB2ENTRY that is not installed.
681	DFHAM4851	DB2CONN not installed because of a security error, or DB2ENTRY not installed because of a security error, or DB2TRAN not installed because of a security error.
682	DFHAM4895	The TST has not been assembled with the migrate option when defining CREATE TSMODEL.
		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	DFHZC6209	Invalid ZC catalog request code *xxxx*.
904	DFHZC6212	Level mismatch with catalog record. DFHBS *xxx*.
905	DFHAM4898 DFHZC5901	Install failed because sufficient storage could not be obtained.
906	DFHZC6200	Could not obtain DWE storage.
907	DFHZC6203	Unable to obtain DWE action-list storage.
908	DFHZC6214	Unable to obtain recovery record storage.
950	DFHZC6202	Pattern *pattern* not valid for builder.
951	DFHZC6204	Illegal subpattern definition *pattern*.
952	DFHZC6205	Illegal subpattern definition *pattern*.
953	DFHZC6206	Pattern *pattern* not valid for destroy.
954	DFHZC6207	Catalog key too long or zero. Pattern *pattern*.
955	DFHZC6213	Recovery record abandoned. Key is *key*.
956	DFHZC6341	Loop or ABEND has been detected in *inmodule* by module *bymodule*.

Table 7. Keywords associated with keyword numbers. CREATE CONNECTION through CREATE DOCTEMPLATE

Keyword number	EXEC CICS CREATE command names				
	CONNECTION	**DB2CONN**	**DB2ENTRY**	**DB2TRAN**	**DOCTEMPLATE**
[1] 1	CONNECTION	DB2CONN	DB2ENTRY	DB2TRAN	DOCTEMPLATE
5	NETNAME				
6	INDSYS	DESCRIPTION	DESCRIPTION	DESCRIPTION	DESCRIPTION
7	SECURITYNAME	DB2ID			TEMPLATENAME
8	(BINDPASSWORD) #	MSGQUEUE1	TRANSID	TRANSID	FILE
9		MSGQUEUE2		ENTRY	TSQUEUE
10	REMOTESYSTEM	MSGQUEUE3			TDQUEUE
11	REMOTENAME	PURGECYCLE			PROGRAM
12	DESCRIPTION				EXITPGM
13	QUEUELIMIT	STATSQUEUE	PROTECTNUM		DDNAME
14	MAXQTIME	TCBLIMIT			MEMBERNAME
15		THREADLIMIT	THREADLIMIT		
16		AUTHID	AUTHID		
17		PLAN	PLAN		
18		PLANEXITNAME	PLANEXITNAME		
19		COMTHREADLIMIT			
20		COMAUTHID			
21		SIGNID			
27	REMOTESYSNET				
97	INSERVICE				
98	AUTOCONNECT	CONNECTERROR			
99	PROTOCOL	NONTERMREL			APPENDCRLF
100	ACCESSMETHOD	STANDBYMODE			
101	SINGLESESS	THREADERROR			
102	DATASTREAM	ACCOUNTREC	ACCOUNTREC		
103	RECORDFORMAT	AUTHTYPE	AUTHTYPE		
104	ATTACHSEC	DROLLBACK	DROLLBACK		
105	BINDSECURITY				
106	CONNTYPE	PRIORITY	PRIORITY		
107	PSRECOVERY	THREADWAIT	THREADWAIT		
108		COMAUTHTYPE			
110	USEDFLTUSER				
111	XLNACTION				

Note:

[1] Keyword number 1 always refers to the first operand of the CREATE command; that is, the resource being created.

Table 8 (Page 1 of 2). Keywords associated with keyword numbers. CREATE ENQMODEL through CREATE MAPSET.

Keyword number	EXEC CICS CREATE command names				
	ENQMODEL	**FILE**	**JOURNALMODEL**	**LSRPOOL**	**MAPSET**
[1] 1	ENQMODEL	FILE	JOURNALMODEL	LSRPOOL	MAPSET
5		(RESSECNUM) #			(RSL) #
6	DESCRIPTION	DSNAME	DESCRIPTION	MAXKEYLENGTH	DESCRIPTION
7	ENQSCOPE	RECORDSIZE	JOURNALNAME	SHARELIMIT	
8	ENQNAME	KEYLENGTH	STREAMNAME	STRINGS	
9		JOURNAL		DATA512	
10		REMOTESYSTEM		DATA1K	
11		REMOTENAME		DATA2K	
12		PASSWORD		DATA4K	
13		LSRPOOLID		DATA8K	
14		STRINGS		DATA12K	
15		DATABUFFERS		DATA16K	
16		INDEXBUFFERS		DATA20K	
17		FWDRECOVLOG		DATA24K	
18		DESCRIPTION		DATA28K	

Keywords in parentheses, such as (RSL), are obsolete but are retained for cross-release compatibility.

Table 8 (Page 2 of 2). Keywords associated with keyword numbers. CREATE ENQMODEL through CREATE MAPSET.

Keyword number	EXEC CICS CREATE command names				
	ENQMODEL	**FILE**	**JOURNALMODEL**	**LSRPOOL**	**MAPSET**
19		NSRGROUP		DATA32K	
20		MAXNUMRECS		LSRPOOLID	
21		CFDTPOOL		DESCRIPTION	
22		TABLENAME		INDEX512	
23				INDEX1K	
24				INDEX2K	
25				INDEX4K	
26				INDEX8K	
27				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		STATUS			STATUS
98		RECOVERY	TYPE		
99	STATUS	OPENTIME			
100		DISPOSITION			RESIDENT
101		ADD			USAGE
102		BROWSE			USELPACOPY
103		DELETE			
104		READ			
105		UPDATE			
106		JNLSYNCREAD			
107		JNLSYNCWRITE			
108		JNLREAD			
109		JNLUPDATE			
110		JNLADD			
111		DSNSHARING			
112		RECORDFORMAT			
113		TABLE			
114		BACKUPTYPE			
115		RLSACCESS			
116		READINTEG			
117		LOAD			
118		UPDATEMODEL			

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 9. Keywords associated with keyword numbers. CREATE PARTITIONSET through CREATE PROGRAM.

Keyword number	EXEC CICS CREATE command names				
	PARTITIONSET	**PARTNER**	**PROCESSTYPE**	**PROFILE**	**PROGRAM**
1 1	PARTITIONSET	PARTNER	PROCESSTYPE	PROFILE	PROGRAM
5	(RSL) #	NETNAME	AUDITLOG	MODENAME	(RSL) #
6	DESCRIPTION	DESCRIPTION	DESCRIPTION	JOURNAL	DESCRIPTION
7		NETWORK	FILE	NEPCLASS	REMOTESYSTEM
8		PROFILE		RTIMOUT	REMOTENAME
9		TPNAME		DESCRIPTION	TRANSID
10		XTPNAME		FACILITYLIKE	
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					
22					
23					
24					
25					
27					
28					
97	STATUS				STATUS
98			STATUS	SCRNSIZE	LANGUAGE
99			AUDITLEVEL	MSGJRNL	RELOAD
100	RESIDENT			MSGINTEG	RESIDENT
101	USAGE			ONEWTE	USAGE
102	USELPACOPY			(PROTECT) #	USELPACOPY
103				DVSUPRT	CEDF
104				INBFMH	DATALOCATION
105				RAQ	EXECKEY
106				LOGREC	
107				PRINTERCOMP	EXECUTIONSET
108				CHAINCONTROL	DYNAMIC
109				UCTRAN	CONCURRENCY

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 10. Keywords associated with keyword numbers. CREATE REQUESTMODEL through CREATE TERMINAL.

Keyword number	EXEC CICS CREATE command names				
	REQUESTMODEL	**SESSIONS**	**TCPIPSERVICE**	**TDQUEUE**	**TERMINAL**
1 1	REQUESTMODEL	SESSIONS	TCPIPSERVICE	TDQUEUE	TERMINAL
5		CONNECTION			
6	DESCRIPTION	SESSNAME	DESCRIPTION	DESCRIPTION	AUTINSTNAME
7	OMGMODULE	NETNAMEQ		BLOCKSIZE	TYPETERM
8	OMGINTERFACE	MODENAME		DATABUFFERS	NETNAME
9	OMGOPERATION	MAXIMUM		DDNAME	CONSOLE
10	TRANSID			DSNAME	REMOTESYSTEM
11		RECEIVEPFX		RECORDSIZE	REMOTENAME
12		RECEIVECOUNT		FACILITYID	MODENAME
13		SENDPFX		TRANSID	PRINTER
14		SENDCOUNT	URM	TRIGGERLEVEL	ALTPRINTER
15		(OPERID) #	PORTNUMBER	USERID	(OPERID) #
16		(OPERPRIORITY) #		INDIRECTNAME	(OPERPRIORITY) #
17		(OPERRSL) #	CERTIFICATE	REMOTENAME	(OPERRSL) #
18		(OPERSECURITY) #	TRANSACTION	REMOTESYSTEM	(OPERSECURITY) #
19		USERID	BACKLOG	SYSOUTCLASS	USERID
20		SENDSIZE		REMOTELENGTH	POOL
21		RECEIVESIZE			TASKLIMIT
22		(TRANSACTION) #	TSQPREFIX		TRANSACTION
23		SESSPRIORITY	IPADDRESS		TERMPRIORITY
24		USERAREALEN			
25		IOAREALEN			
26					SECURITYNAME
27		NEPCLASS			(BINDPASSWORD) #
28		DESCRIPTION			DESCRIPTION
29					NATLANG
30					CONSNAME
33					REMOTESYSNET
97		(INSERVICE) #			INSERVICE
98		AUTOCONNECT		TYPE	PRINTERCOPY
99		BUILDCHAIN		DISPOSITION	ALTPRINTCOPY
100		PROTOCOL		ERROROPTION	AUTINSTMODEL
101		RELREQ		OPENTIME	
102		DISCREQ		RECORDFORMAT	ATTACHSEC
103		RECOVOPTION		BLOCKFORMAT	BINDSECURITY
104		(RECOVNOTIFY) #	STATUS	REWIND	USEDFLTUSER
105			SSL	TYPEFILE	
106				ATIFACILITY	
107				RECOVSTATUS	
108				WAITACTION	
109				PRINTCONTROL	
110				WAIT	

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 11 (Page 1 of 2). Keywords associated with keyword numbers. CREATE TRANCLASS through CREATE TYPETERM.

Keyword number	EXEC CICS CREATE command names			
	TRANCLASS	**TRANSACTION**	**TSMODEL**	**TYPETERM**
1 1	TRANCLASS	TRANSACTION	TSMODEL	TYPETERM
5	MAXACTIVE	(RSL) #		DEVICE
6	DESCRIPTION	PROGRAM	DESCRIPTION	TERMMODEL
7	PURGETHRESH	TWASIZE	PREFIX	SESSIONTYPE
8		PROFILE	POOLNAME	
9		PARTITIONSET	REMOTESYSTEM	LDCLIST
10		REMOTESYSTEM	REMOTEPREFIX	DEFSCREEN
11		REMOTENAME	XPREFIX	
12		PRIORITY	XREMOTEPFX	ALTSCREEN
13		(TCLASS) #		
14		TASKREQ		CGCSGID
15		XTRANID		
16		DTIMOUT		SENDSIZE
17		(TRANSEC) #		RECEIVESIZE
18		TRPROF		LOGMODE
19		(PRIMEDSIZE) #		PAGESIZE
20		ALIAS		
21		DESCRIPTION		ALTPAGE
22		TPNAME		
23		XTPNAME		ALTSUFFIX
24		TRANCLASS		USERAREALEN
25		RUNAWAY		IOAREALEN
26		WAITTIME		
27				NEPCLASS
28				DESCRIPTION
29		BREXIT		
30				
33				
97		STATUS		
98		LOCALQ		AUTOCONNECT
99		(INDOUBT) #	LOCATION	SHIPPABLE
100		RESTART	RECOVERY	APLKYBD
101		SPURGE	SECURITY	APLTEXT
102		TPURGE		AUDIBLEALARM
103		DUMP		COLOR
104		(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		WAIT		VALIDATION
116		ACTION		FORMFEED
117		ROUTABLE		HORIZFORM
118				VERTICALFORM
119				TEXTKYBD
120				TEXTPRINT
121				QUERY
122				OUTLINE
123				SOSI
124				BACKTRANS
125				ASCII
126				BRACKET

Keywords in parentheses, such as (RSL), are obsolete but are retained for cross-release compatibility.

Table 11 (Page 2 of 2). Keywords associated with keyword numbers. CREATE TRANCLASS through CREATE TYPETERM.

Keyword number	EXEC CICS CREATE command names			
	TRANCLASS	**TRANSACTION**	**TSMODEL**	**TYPETERM**
127				FMHPARM
128				OBOPERID
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				RECOVOPTION
145				RECOVNOTIFY
146				XRFSIGNOFF
147				(LOGEMODECOM) #

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.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Sending your comments to IBM

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, to this address:

  Information Development Department (MP095)
  IBM United Kingdom Laboratories
  Hursley Park
  WINCHESTER,
  Hampshire
  SO21 2JN
  United Kingdom

- 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™: HURSLEY(IDRCF)
  - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Program Number: 5655-147

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC33-1689-02

Spine information:

CICS TS for OS/390 CICS System Programming Reference *Release 3*