



iSeries

CICS for iSeries Administration and Operations Guide

Version 5

SC41-5455-00





@server

iSeries

CICS for iSeries Administration
and Operations Guide

Version 5

SC41-5455-00

Note!

Before using this information and the product it supports, be sure to read the information in "Notices" on page 355.

First Edition (September 2002)

This edition applies to Version 5 of the IBM licensed program CICS Transaction Server for iSeries (also known as CICS for iSeries or CICS/400), program number 5722-DFH, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. This edition applies only to reduced instruction set computer (RISC) systems.

This edition replaces *Administration and Operations Guide* for CICS/400, SC33-1387-01.

© **Copyright International Business Machines Corporation 2002. All rights reserved.**

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

Contents

About CICS for iSeries Administration and Operations Guide (SC41-5455) . . . vii

Who should read this book	vii
Conventions and terminology used in this book	viii
Prerequisite and related information	viii
CICS/400 library	viii
Books from related libraries	ix
How to send your comments	ix

Summary of Changes xi

Part 1. Introduction 1

Chapter 1. Introducing CICS® for iSeries™ 3

What is CICS for iSeries?	3
What is CICS/400?	4
Server support for CICS clients	5
CICS/400 architecture	5
OS/400 facilities used by CICS/400.	6
Presentation to the end user	7
Operating interfaces	8
Data management	10
Resource management.	12
Intersystem communication	13
Transaction management	14
Application development.	15
Portability and migration.	16
Porting application programs	16
Release-to-release compatibility.	17
Summary	18
Where next?	18

Chapter 2. User-based pricing. 19

What is user-based pricing?	19
How does the license manager count the number of users?	19
What is a user?	19
What is not a user?.	20

Chapter 3. Installing CICS/400. 21

Installation	21
Migration	21
Upward migration	21
INZCICS	22
Downward migration	22
Installation verification procedure	23
Installing the sample code from tape	23
The IVP process	24
Step 1. Creating a working copy of QCICSSAMP	24
Step 2. Adding libraries to the OS/400 library list	24
Step 3. Translating the BMS map	25
Step 4. Translating the program.	26

Step 5. Creating the recoverable/nonrecoverable resources	27
Step 6. Journaling the CICS/400 recoverable resources	28
Step 7. Defining the CICS/400 resources.	28
Step 8. Creating the CL program	28
Step 9. Starting the CICS/400 control region	33
Step 10. Starting the sample transaction	33
DEFICSRGN	34
DEFICSRGN	34
Running DEFICSRGN	35
Autodefined resources.	36
Deleting the autodefined control region	38

Part 2. System administration 39

Chapter 4. Defining resources. 41

What is resource definition?	41
Gathering preliminary information	41
Resource definition tables	42
Resource definition architecture.	42
Creating resource definitions	45
Defining and maintaining group definitions	45
What is a resource definition group?	46
Using the resource definition CL commands	46
Specifying which groups are used	47
Recovering groups	48
Installing a group	48
Using the ADDCICSGLT command	48
Using the INSCICSGRP command.	48
Using the CEDA Install group function	49
Defining and maintaining individual resource definitions.	51
How to use the resource definition commands.	52
Entering CL commands from the command line	52
Using the prompt screens.	52
Entering resource and transaction identifiers	54
Getting further parameters	54
Getting help	54
Using WRKCICSxxx commands	54
Using CEDA resource definition online	58

Chapter 5. Defining a basic control region 61

Defining the system initialization table (SIT)	61
Defining temporary storage and transient data files	63
File characteristics	63
Creating the physical files	63
Defining supplied transactions	65
Setting default wait times.	67
Setting job priorities	67
Storage considerations.	67
Control region storage objects	68
Shell storage objects	72

Summary	74
Chapter 6. Security requirements for CICS/400	75
Signon security and display station security	75
Initial menu and initial program security	76
OS/400 command security	76
Limited capability checking	76
Resource checking of command program objects	76
Resource security	77
Supplied transactions	77
User transactions	77
Application example	77
Adoptive authority	78
Communication security	78
User profile and group profile security	78
Chapter 7. Defining your own control region	81
OS/400 data management considerations	82
Transaction, program, and map considerations	82
Example PCT definitions	83
Example PPT definitions	83
Temporary storage considerations	84
Recoverable and nonrecoverable TS queues	84
Example TST definitions	85
Transient data considerations	85
Extrapartition transient data	86
Recoverable intrapartition TD files	86
Example DCT definitions	86
Defining a CSMT log destination	87
File control considerations	88
Supported file types	88
Automatic file closure	89
Recoverable files	90
Example FCT definitions	90
Interval control considerations	94
Timer-related tasks	94
Protected interval control start requests	94
Interval control elements	95
Acquiring terminals	95
Interval control batch shells	95
Request identifiers	96
Start requests and autoinstalled terminals	96
Journal control considerations	96
Journal records	96
Journal output synchronization	97
Creating CICS/400 user journals	97
Example creation of CICS/400 journal	99
Example JCT definitions	100
Device considerations	100
Defining local CICS terminals	100
Example TCT definitions	100
Defining remote systems	101
Example TCS entry	101
Remote resource considerations	102
Defining remote resources	102
Defining remote terminals	102
Example TCT definitions for remote and shippable terminals	102

Printer spooling considerations	102
Trace considerations	104
Internal trace	104
Printing CICS/400 trace	104
Recovery considerations	104
Recoverable file commitment control requirements	104
OS/400 commitment control and logical unit of work (LUW)	105

Chapter 8. Administering the control region	107
Introduction to the control region	107
Control region processing	108
Control region initialization	108
Control region runtime processing	109
Control region shutdown	111
Starting a control region (STRCICS)	112
STRCICS	112
Setting the STRTYPE parameter	114
Ending a control region (ENDCICS)	118
ENDCICS	118

Chapter 9. Administering a CICS/400 shell	121
Introduction to the CICS/400 shell	121
Shell objects	121
Shell processing	122
Shell initialization	122
Runtime processing	123
Shell shutdown	124
Starting a user shell (STRCICSUSR)	125
STRCICSUSR	125
Ending a user shell (ENDCICSUSR)	127
ENDCICSUSR	128
CESF—CICS sign-off transaction	129

Part 3. CICS resources 131

Chapter 10. Defining resources-reference information	133
Interpreting the syntax diagrams	133
Managing groups	135
Using the CRTCICSGRP command	135
Using the CHGCICSGRP command	137
Using the DLTCICSGRP command	138
Using the INSCICSGRP command	138
Using the SAVCICSGRP command	140
Using the WRKCICSGRP command	141
Managing data conversion resource definitions	142
Using the ADDCICSCVT command	143
Using the CHGCICSCVT command	149
Using the DSPCICSCVT command	155
Using the RMVCICSCVT command	157
Using the WRKCICSCVT command	159
Managing destination definitions	160
Using the ADDCICSDCT command	161
Using the CHGCICSDCT command	167
Using the DSPCICSDCT command	173

Using the RMVCICSDCT command	175
Using the WRKCICSDCT command	176
Managing file resource definitions	177
Using the ADDCICSFCT command	177
Using the CHGCICSFCT command	183
Using the DSPCICSFCT command	189
Using the RMVCICSFCT command	191
Using the WRKCICSFCT command	192
Managing group list resource definitions	193
Using the ADDCICSGLT command	194
Using the DSPCICSGLT command	196
Using the RMVCICSGLT command	198
Using the WRKCICSGLT command	200
Managing journal resource definitions	201
Using the ADDCICSJCT command	202
Using the CHGCICSJCT command	206
Using the DSPCICSJCT command	209
Using the RMVCICSJCT command	210
Using the WRKCICSJCT command	212
Managing transaction definitions	213
Using the ADDCICSPCT command	214
Using the CHGCICSPCT command	218
Using the DSPCICSPCT command	222
Using the RMVCICSPCT command	223
Using the WRKCICSPCT command	225
Managing program definitions	226
Using the ADDCICSPPT command	227
Using the CHGCICSPPT command	231
Using the DSPCICSPPT command	235
Using the RMVCICSPPT command	236
Using the WRKCICSPPT command	238
Managing system initialization resource definitions	239
Using the ADDCICSSIT command	239
Using the CHGCICSSIT command	248
Using the DSPCICSSIT command	258
Using the RMVCICSSIT command	259
Using the WRKCICSSIT command	260
Managing system resource definitions	260
Using the ADDCICSTCS command	261
Using the CHGCICSTCS command	265
Using the DSPCICSTCS command	269
Using the RMVCICSTCS command	270
Using the WRKCICSTCS command	272
Managing terminal resource definitions.	273
Uppercase conversion of terminal input	273
Datastream compression.	274
Using the ADDCICSTCT command	275
Using the CHGCICSTCT command	282
Using the DSPCICSTCT command	291
Using the RMVCICSTCT command	292
Using the WRKCICSTCT command	294
Managing temporary storage resource definitions	295
Using the ADDCICSTST command	296
Using the CHGCICSTST command	298
Using the DSPCICSTST command	300
Using the RMVCICSTST command	301
Using the WRKCICSTST command	302

Chapter 11. Autoinstall for terminal definitions. 305

How autoinstall terminal identifiers are generated 307

Defining an autoinstall model terminal using masking	308
Using an autoinstall control program	309
The CICS/400-supplied autoinstall control program	309
How does CICS/400 know to call an autoinstall control program?	309
Creating a PPT definition for AEGTCACP.	309
Translating the autoinstall control program	310
Security	311
Batch shell considerations	311
Storage considerations	311
CEMT INQUIRE/SET PROGRAM and EXEC CICS INQUIRE/SET PROGRAM.	312
Turning off autoinstall in an active CICS system	312
Attempting to install an AEGTCACP PPT entry defined as remote	312
Writing your own autoinstall program	312
Autoinstalling terminal definitions	313
Returning information to CICS/400	315
Selecting the autoinstall model	316
Setting the terminal name	316
CICS/400 action on return from the control program	317
Deleting autoinstalled terminal definitions.	317
Naming your control program.	318
Testing and debugging your control program	318
Customizing the sample program	319

Part 4. CICS-supplied transactions 321

Chapter 12. Introduction to CICS/400-supplied transactions . . . 323

Syntax notation.	323
Online help	323
Levels of access to supplied transactions	323
Role of the system administrator	324

Chapter 13. CEMT—CICS master terminal transaction 325

Using the CEMT transaction	325
Entering a CEMT command	326
Minimum abbreviation of keywords.	326
Selecting resources	326
Family of resources	327
List of resource identifiers	327
Displaying options and syntax (the ? key).	328
Scrolling data (+ sign)	328
Blank fields in a display.	328
Using the CEMT screens	329
Detecting unprotected fields (the tab key)	330
Program function (PF) keys.	330
Overtyping a CEMT INQUIRE display	331
CEMT DISCARD command	332
CEMT DISCARD	332
CEMT INQUIRE and SET commands	333
CEMT INQUIRE AUTINSTMODEL	333
CEMT INQUIRE SET AUXTRACE	334
CEMT INQUIRE SET CONNECTION	335
CEMT INQUIRE SET FILE.	336

CEMT INQUIRE SET INTTRACE	338
CEMT INQUIRE SET JOURNALNUM.	339
CEMT INQUIRE SET NETNAME	340
CEMT INQUIRE SET PROGRAM	340
CEMT INQUIRE SYSTEM	342
CEMT INQUIRE SET TASK	343
CEMT INQUIRE SET TDQUEUE	344
CEMT INQUIRE SET TERMINAL	346
CEMT INQUIRE SET TRANSACTION.	348
CEMT PERFORM commands	348

Chapter 14. CRTE—CICS routing transaction	351
--	------------

Part 5. Appendixes	353
Notices	355
Programming Interface Information	357
Trademarks	357
Glossary	359
Index	369

About CICS for iSeries Administration and Operations Guide (SC41-5455)

This book contains information relevant to the operation and administration of an IBM CICS/400 system. It provides an overview of CICS/400, and then addresses the following major topics:

- Characteristics and architecture of a CICS/400 system
- CICS resource definition
- Operating procedures, including starting and ending CICS/400 shells and control regions
- CICS-supplied transactions, including the master terminal transaction CEMT, that help you administer your CICS/400 system

This book contains information relevant to the operation and administration of an IBM CICS/400 system. It provides an overview of CICS/400, and then addresses the following major topics:

Study of these topics should provide the information required to set up and administer CICS on an iSeries.

Who should read this book

This book is intended for the person responsible for CICS/400 systems administration. The iSeries system is intended primarily for an end-user environment. In keeping with this objective, a comprehensive systems programming function is not necessary to support the installation.

Application programmers and analysts should not require the information in this book; however, they may like to browse it to gain general background knowledge.

The reader is assumed to have general data processing knowledge with an understanding of CICS, and an understanding of or access to information about the iSeries. The CICS/400 library does not attempt to teach the fundamentals of CICS. Extensive systems programming experience should not be necessary.

If you are responsible for setting up a control region, you should review the *CICS for iSeries Application Programming Guide* for general knowledge of the following:

- The *CICS management functions* gives an overview of the functionality of the CICS components.
- The *Files and databases* covers how VSAM is emulated on the OS/400.
- The *System programming* contains details of the EXEC CICS commands that complement the CEMT transaction.

The book is organized into four parts:

- Part 1, "Introduction", contains an overview of CICS/400, its architectural design, and how to install it on your iSeries.
- Part 2, "System administration", describes tasks such as defining a control region and administering control regions and user shells.
- Part 3, "CICS resources", contains reference information on the commands used to define resources on the OS/400, and to define autoinstall terminals.

- Part 4, “CICS-supplied transactions”, describes the supplied transactions that a systems administrator might use.

Conventions and terminology used in this book

The term ‘CICS’ refers to ‘CICS/400’, that is ‘CICS for OS/400’, unless otherwise stated.

MB equals 1 048 576 bytes.

KB equals 1024 bytes.

Prerequisite and related information

Use the iSeries Information Center as your starting point for looking up iSeries technical information.

You can access the Information Center two ways:

- From the following Web site:
<http://www.ibm.com/eserver/series/infocenter>
- From CD-ROMs that ship with your Operating System/400 order:
iSeries Information Center, SK3T-4091-02. This package also includes the PDF versions of iSeries manuals, *iSeries Information Center: Supplemental Manuals*, SK3T-4092-01, which replaces the Softcopy Library CD-ROM.

The iSeries Information Center contains advisors and important topics such as Java, TCP/IP, Web serving, secured networks, logical partitions, clustering, CL commands, and system application programming interfaces (APIs). It also includes links to related IBM Redbooks and Internet links to other IBM Web sites such as the Technical Studio and the IBM home page.

With every new hardware order, you receive the *iSeries Setup and Operations CD-ROM*, SK3T-4098-01. This CD-ROM contains IBM @server iSeries Access for Windows and the EZ-Setup wizard. iSeries Access offers a powerful set of client and server capabilities for connecting PCs to iSeries servers. The EZ-Setup wizard automates many of the iSeries setup tasks.

CICS/400 library

These books form the CICS/400 library that is delivered with the product:

CICS for iSeries Administration and Operations Guide, SC41-5455-00

This guide gives introductory information about CICS/400. It then provides information about system and resource definition, setup of a system, and operator commands.

CICS for iSeries Application Programming Guide, SC41-5454-01

This manual provides programming guidance information, in narrative form with examples. This is followed by the reference section describing the syntax and use of each command.

CICS for iSeries Intercommunication, SC41-5456-00

This manual describes the CICS/400 side of communication between CICS systems running on different platforms. There is a similar manual for each CICS platform.

CICS for iSeries Problem Determination, SC41-5453-00

This manual provides guidance in problem determination for users of CICS/400.

CICS Family: Interproduct Communication, SC34-6030-00

This manual, which is also part of the libraries of the other CICS family members, gives an overview of communication between CICS systems running on different platforms.

CICS Family: API Structure, SC33-1007-02

This manual, which is also part of the libraries of the other CICS family members, gives a quick reference to the level of support that each member of the CICS family gives to the CICS application programming interface. It is designed for customers and software vendors developing applications able to run on more than one CICS platform and porting applications from one platform to another.

Books from related libraries

Some other manuals may be useful to the readers of this book.

CICS intercommunication manuals

CICS on System/390 Intercommunication Guide (refer to the Intercommunication Guide for your CICS on System/390 product)

CICS for OS/2 Intercommunication

CICS on Open System Intercommunications Guide

iSeries manuals

Communications Configuration, SC41-5401-00

ICF Programming, SC41-5442-00

Communications Management, SC41-5406-02

APPC Programming, SC41-5443-00

Work Management, SC41-5306-03

Globalization topic in the iSeries Information Center

Software Installation, SC41-5120-06

Other books

SNA Formats, GA23-3136

SNA LU 6.2 Peer Protocols Reference , SC31-6808

CPI Communications Reference, SC26-4399

AIX SNA Server/6000 V2R1 User's Guide, SC31-7002

How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other iSeries documentation, fill out the readers' comment form at the back of this book.

- If you prefer to send comments by mail, use the readers' comment form with the address that is printed on the back. If you are mailing a readers' comment form from a country other than the United States, you can give the form to the local IBM branch office or IBM representative for postage-paid mailing.
- If you prefer to send comments by FAX, use either of the following numbers:
 - United States, Canada, and Puerto Rico: 1-800-937-3430
 - Other countries: 1-507-253-5192

- If you prefer to send comments electronically, use one of these e-mail addresses:
 - Comments on books:
RCHCLERK@us.ibm.com
 - Comments on the iSeries Information Center:
RCHINFOC@us.ibm.com

Be sure to include the following:

- The name of the book or iSeries Information Center topic.
- The publication number of a book.
- The page number or topic of a book to which your comment applies.

Summary of Changes

Changes for this release include support for connectivity of CICS Universal Clients and the CICS Transaction Gateway to CICS for iSeries using TCP/IP connectivity.

Part 1. Introduction

Chapter 1. Introducing CICS® for iSeries™	3
What is CICS for iSeries?	3
What is CICS/400?	4
Server support for CICS clients	5
CICS/400 architecture	5
OS/400 facilities used by CICS/400.	6
Presentation to the end user	7
Operating interfaces	8
Data management	10
Resource management.	12
Intersystem communication	13
Transaction management	14
Application development.	15
Portability and migration.	16
Porting application programs	16
Release-to-release compatibility.	17
Summary	18
Where next?	18
Chapter 2. User-based pricing	19
What is user-based pricing?	19
How does the license manager count the number of users?	19
What is a user?	19
What is not a user?.	20
Chapter 3. Installing CICS/400	21
Installation	21
Migration	21
Upward migration	21
INZCICS	22
Downward migration	22
Installation verification procedure	23
Installing the sample code from tape	23
The IVP process	24
Step 1. Creating a working copy of QICSSAMP	24
Step 2. Adding libraries to the OS/400 library list	24
Step 3. Translating the BMS map	25
Step 4. Translating the program.	26
Step 5. Creating the recoverable/nonrecoverable resources	27
Step 6. Journaling the CICS/400 recoverable resources	28
Step 7. Defining the CICS/400 resources.	28
Step 8. Creating the CL program	28
Step 9. Starting the CICS/400 control region	33
Step 10. Starting the sample transaction	33
DEFCICSRGN	34
DEFCICSRGN	34
Running DEFCICSRGN	35
Autodefined resources.	36
Deleting the autodefined control region	38

Chapter 1. Introducing CICS® for iSeries™

This chapter introduces you to CICS/400 and places it in the iSeries environment. It introduces some important CICS concepts such as resource management and intersystem communication, and some of the facilities provided to assist application programmers to develop programs or migrate existing programs to CICS/400 from other CICS platforms. It also introduces the concept of distributed processing and how CICS/400 helps you to present data to the end user and control your transaction. The topics covered are:

- “What is CICS for iSeries?”
- “What is CICS/400?” on page 4
- “OS/400 facilities used by CICS/400” on page 6
- “Presentation to the end user” on page 7
- “Operating interfaces” on page 8
- “Data management” on page 10
- “Resource management” on page 12
- “Intersystem communication” on page 13
- “Transaction management” on page 14
- “Application development” on page 15
- “Portability and migration” on page 16

It is assumed that you are familiar with Operating System/400® (OS/400) and that you have some CICS knowledge.

What is CICS for iSeries?

Customer Information Control System (CICS) is an online transaction processing (OLTP) system that provides:

- Control of concurrently-running applications serving many online users
- The functions required by your application programs for communicating with remote and local terminals and subsystems
- Control of files and databases in conjunction with the various IBM data access methods
- The ability to communicate with other CICS systems and database systems, using CICS intersystem communication (ISC)
- An application programming interface that is invoked using EXEC CICS commands

CICS is available on a number of different platforms. Mainframe-based CICS includes:

- CICS Transaction Server for z/OS™ Version 2
- CICS Transaction Server for OS/390® Version 1
- CICS/ESA Version 4
- CICS Transaction Server for VSE/ESA™
- CICS/VSE® Version 2

CICS for Open Systems includes:

- TXSeries™ Version 5.0 for Multi-platforms, which contains:

- CICS for AIX[®]
- CICS for HP-US
- CICS for Sun Solaris
- CICS for Windows NT[®]
- TXSeries Version 4.3 for AIX (which contains CICS for AIX)
- TXSeries Version 4.3 for Sun Solaris (which contains CICS for Sun Solaris)
- TXSeries Version 4.3 for WIndows NT (which contains CICS for Windows NT)
- TX Series Version 4.2 for HP-UX (which contains CICS for HP-UX)

CICS for OS/2[®] runs on IBM compatible personal computers. CICS for iSeries runs under the OS/400[®] operating system on iSeries and AS/400[®] systems.

The key features of the CICS family, allowing significant source compatibility and access to data and services across CICS systems, are:

- Provision of transaction processing services
- Common application programming interface
- Intersystem communication (ISC)
- Access to a range of database products
- Data integrity for all users

What is CICS/400?

CICS/400 brings to the iSeries range of computers the benefits of mainframe CICS, but it has been specially designed for the easy-to-use OS/400 environment. CICS/400 extends the OS/400 transaction processing capability by providing CICS functions through the CICS application programming interface, building on the native OS/400 function. It is a CICS family product, having many of the familiar features of mainframe CICS, but adapted to be familiar to the OS/400 user as well. The external interfaces for the administration tasks required for CICS/400 have the look and feel of OS/400.

CICS/400 can run in two environments:

- **Stand-alone system**

In this type of environment, new applications can be developed on the OS/400, taking full advantage of existing CICS skills. Existing mainframe CICS applications can be run on the OS/400 provided that they conform to the requirements of CICS/400. See “Portability and migration” on page 16 for guidelines.

- **Networked system**

Over a network of interconnected CICS systems, you can develop distributed applications. Typically, you would choose to place the presentation logic close to the user, and the business logic close to the data it requires. The systems themselves could all be CICS/400s, or may include other CICS family members, for example CICS for MVS/ESA, CICS for OS/2, or CICS workstation-based clients. You could use CICS/400 as a front-end system connected to CICS on a mainframe, a CICS workstation-based client as a front end to CICS/400, or even a mainframe CICS as a front end to CICS/400. Each processor—mainframe, iSeries, or workstation—could then be used for the purposes for which it is best suited. Its suitability may be by virtue of its proximity to data or users, or by virtue of the interface it is able to provide as a front-end processor.

In a distributed processing environment, you can also place your data in accordance with your business needs. Data relevant to only one part of your

business can be placed on the user's computer, thereby improving response times and reducing communications costs. Databases that are going to be used by your whole business might be placed on a mainframe, or a centralized OS/400 system.

With the CICS transaction processing family, you can locate programs and data on the most appropriate processor. When a transaction is started, CICS responds to the request, and the end user is unaware of the location of the data or the programs being run. This is achieved using CICS intersystem communication (ISC).

Server support for CICS clients

CICS/400 provides support for those CICS clients that can connect to the iSeries over Advanced Program-to-Program Communication (APPC) or TCP/IP communications. These include:

- AIX
- Microsoft® Windows
- OS/2
- Solaris
- HP-UX
- Linux 390

CICS client applications are written using programming interfaces provided on the workstation:

- External Presentation Interface (EPI)
- External Call Interface (ECI)

The EPI allows existing CICS applications to exploit user-friendly graphical user interfaces (GUIs) on the workstation without the need for changes.

The ECI allows the design of fully distributed applications. Typically the business logic is kept on a CICS server, while the presentation logic can be implemented on the workstation to take advantage of available GUIs.

CICS/400 architecture

A CICS/400 system is a collection of OS/400 jobs. One type of job, a *control region*, provides the control, scheduling, and work management mechanisms necessary to coordinate all the shared resources of a CICS environment. The other type of job, a *shell*, provides COBOL/400 and ILE C programs with an interface to the CICS-managed resources and maintains the application environment required to execute CICS transactions.

Before you can run CICS transactions you must start a CICS/400 control region and at least one CICS/400 shell associated with that control region. You may start more than one control region, each of which will independently manage its own set of resources and provide services to a set of shells. Usually a control region will have many shells associated with it, but a shell can only be associated with one control region.

Transactions run as CICS/400 tasks within a shell. The separation of duties between a control region and its shells, together with the separation of shared and nonshared storage within the control region and the shells, provides protection to the overall CICS system in the event of failure. If a shell ends, either normally or

abnormally, the other shells and the control region should be unaffected. Similarly, if a control region ends, other control regions on the same iSeries system are unaffected.

OS/400 facilities used by CICS/400

The iSeries is an object-oriented system. This means that files, programs, user spaces, and data queues are all treated as objects on the iSeries.

CICS/400 does not preclude the use of functions already present in native iSeries systems. Applications from the two systems are complementary, and can be run side by side. Applications can include both EXEC CICS and native CL commands. Where appropriate, CICS/400 relies on OS/400 functions. For example, OS/400 has built-in security mechanisms for allowing or controlling access to objects on the iSeries. CICS/400 relies on the OS/400 security mechanisms to control access to CICS-defined objects (for example, programs, files, or data queues).

CICS/400 makes use of some of the facilities of OS/400 to provide a product that is familiar to OS/400 users. These are:

- Installation

CICS/400 is an iSeries Licensed Program and is distributed on the Licensed Program media . CICS/400 is installed and operated using the standard OS/400 procedure for licensed programs. An installation verification procedure (IVP) and a sample transaction ACCT, which are included as part of CICS/400, may be run to establish that your CICS/400 system has been installed correctly.

- Security

CICS/400 uses the security facilities of OS/400 to provide system-level and object-level security. There is no signon security to CICS/400; signon security is established when the user signs on to the iSeries.

OS/400 Advanced Program-to-Program Communication (APPC) support also provides LU6.2 bind-time security and user security.

- Work management

You use the work management facilities of OS/400 to manage the control region and the shells in the same way as any other job. For example, you need to set up a subsystem description to define the CICS/400 subsystem, and job descriptions to describe the control region and each shell, which run as jobs within the subsystem. Also, each user of your CICS/400 system will require a user profile, to define what the user is allowed to do using CICS/400.

- File control

CICS/400 uses the native OS/400 file and commitment control facilities for ensuring the integrity of data files, including record locking and rollback control. Native OS/400 file objects are opened by CICS user shells. CICS shells can share these resources and use native system services to maintain data integrity.

- Commitment control

CICS/400 builds upon OS/400 commitment control to extend integrity and recoverability to CICS/400 resources such as temporary storage queues. By using “protected conversations” multiple CICS systems can participate to ensure integrity and recoverability across all parts of distributed CICS transactions. This protection uses two-phase commit procedures and sync-level 2 conversations.

- Performance

Performance management for CICS/400 is provided through a combination of commands within the OS/400 licensed program and the Performance Tools for iSeries licensed program. This support provides:

- Functions to collect, measure, and report system performance data at various levels of detail
 - Functions to assist with performance tuning, work load scheduling, application tuning, capacity planning, and iSeries system sizing
 - The capability to display or print data in graphic or tabular form
 - The capability to extract the data through DB2[®] Query Manager and SQL Development Kit for iSeries licensed program functions
- CL commands

All the CL commands that you can use to administer a CICS/400 control region are available on the CMDCICS menu. To access this menu, type GO on the command line of the iSeries Main Menu. The **Go to Menu (GO)** is displayed. In the **Menu** field, type CMDCICS and press Enter. All CICS/400 commands are displayed in alphabetical order. Figure 1 shows part of the display. To use one of the commands, type either the number or the command itself on the command line and the first screen for the command is displayed. Each of the commands on the list is described either in this book or in the *CICS for iSeries Application Programming Guide*.

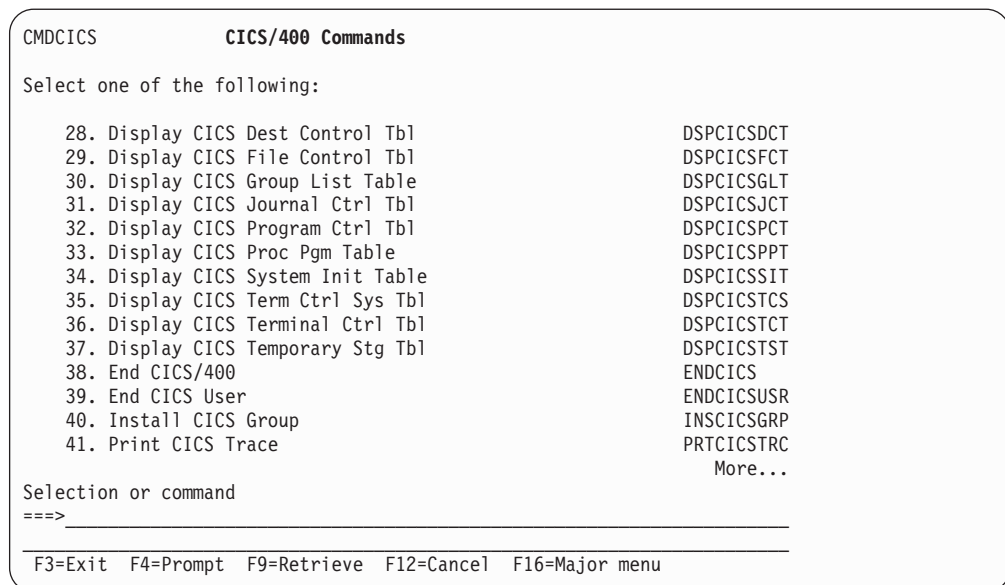


Figure 1. Example CMDCICS screen

Presentation to the end user

One of the principal features of a transaction processing system is that it puts the control of the system in the hands of the end user. Typically, as an end user, you see a series of screens on your workstation that asks for input data or allows you to request transaction services. If necessary, your request may be routed to another CICS system, using the communication facilities, without your being aware of the location of the data or services provided.

- Creating screen displays

When you send data to a screen, you expect to predefine the positioning of the data, the length of the data, and the control information, for example, when to start a new line in printer output. The presentation part of your application

program would contain the layout of the data with all the spaces in between for proper alignment. This means a larger program and additional data (all the spaces) being transmitted.

For CICS/400, the Basic Mapping Support (BMS) facility is used instead of DDS to produce alphanumeric screens. It also provides device independence by removing the need for device knowledge from the application programmer. You can then change the output device without the need for reprogramming your application.

You create your output format, or **map**, to define the characteristics of the map and its data fields. The application program simply moves data to the fields. BMS transmits only data, not spaces, which cuts down on transmission time.

Mainframe applications using BMS can be transferred to run under OS/400 with virtually no change, provided that they conform to the CICS/400 application programming interface, and the level of BMS supported by CICS/400.

BMS supports 3270 and 5250 display devices, both standard and large display models, and ASCII terminals.

- Sending data to a terminal

You can send data to a terminal using either BMS or terminal control commands. When BMS commands are used, CICS/400 generates a 3270 or 5250 data stream, depending on the device definition. When terminal control commands are used, the application provides a 3270 data stream. If the device is defined as a 5250, CICS/400 uses the OS/400 translation facilities to translate the data stream.

The terms *BMS commands* and *terminal control* commands are convenient ways of referring to the CICS/400 commands used to handle data input and output. These commands are part of the standard CICS/400 application programming interface and are used in the same way. They are described, with the other commands, in the *CICS for iSeries Application Programming Guide*.

- Printing

CICS/400 supports 3270 buffered printers and SCS printers.

Operating interfaces

Operator interaction with CICS/400 is through either CICS-supplied transactions or OS/400 commands. This section describes some of the CICS/400 and OS/400 facilities available to enable you to communicate with your CICS/400 system.

- OS/400 CL commands

CL commands are available for a number of operating tasks. It has already been mentioned that there are CL commands for defining and working with resources. CL commands are also available for starting and ending a control region, starting and ending shells, compiling programs, formatting and printing trace data, and many others. These commands have a format that will be familiar to you as an OS/400 user and display prompt screens that look like native OS/400 screens.

- Signing on

CICS/400 does not have its own signon procedures. All users must first sign on to the OS/400.

- Starting a transaction

Transactions are usually started by entering a transaction identifier of up to four characters on an interactive shell display. CICS initiates the associated program

object by referring to the resource definition program control table. See Chapter 4, “Defining resources” on page 41 and “Managing transaction definitions” on page 213.

There are many other ways of starting a transaction. For example, the shell startup procedure and a transaction identifier can be included in a user’s profile, or you can enter a transaction identifier with the shell start command, or the applications that a user can run could be presented on a menu. Transactions may also be started by automatic transaction initiation (ATI) or by intersystem communication requests.

- CICS-supplied transactions

CICS/400 includes a number of supplied transactions that help you to run and administer your system. The resource definition transaction CEDA is introduced in “Resource management” on page 12, and the development transactions CECL, CEBR, and CEDF are introduced in “Application development” on page 15. In addition to these, there are the following transactions that are supplied as a standard part of every CICS/400 system:

- The master terminal transaction CEMT

The master terminal transaction CEMT allows you to control the system and its resources. You use CEMT while CICS/400 is running, so you can change and adjust the system, or correct a problem, without taking down your CICS/400 system. With CEMT, you can:

- Control the number of tasks, or the number of certain types of tasks, that are running at any given time
- Purge a particular task from the system
- Enable or disable a particular transaction
- Enable or disable a file
- Start and stop tracing
- Install a newly compiled copy of an application program
- End a control region
- Perform a CICS/400 dump

For more information about CEMT, see Chapter 13, “CEMT—CICS master terminal transaction” on page 325.

CEMT is a powerful tool and its use can significantly affect your system and its users. You should ensure that you give this transaction adequate security protection in a production system.

- The routing transaction CRTE

The routing transaction CRTE provides one way of running a transaction that is defined to another control region, either on the same CICS system or on a remote CICS system. The remote system may be on the same or another iSeries platform, or on a different platform.

- The sign-off transaction CESF

The sign-off transaction CESF is used to shut down a shell from within CICS/400.

In addition, there is a suite of supplied internal-transactions for supporting intersystem communication and CICS clients.

- Messages

CICS/400 messages are displayed on the job logs of CICS/400 jobs, in 5250 terminal job logs, on the QSYSOPR system operator queue, or in a special

transient data queue called CSMT. Completed job logs can be viewed in the OS/400 spooler, using the WRKSPLF command.

- Online help
Online help is available for all fields on the resource definition panels, as you would expect for an OS/400 system. The online help is invoked in the same way as for any OS/400 system. Context-sensitive help is available also for each CICS/400 message.
- National Language Support
CICS/400 is enabled for full National Language Support (NLS).

Data management

CICS/400 uses the OS/400 Data Management facilities to control access to physical files for CICS/400 applications.

- Supported file types
CICS/400 emulates the VSAM support required by CICS file control using the OS/400 file structure. CICS/400 uses the following file types:
 - Key-sequenced (KSDS)
A KSDS has each of its records identified by a key; that is, a field at a predefined position in the file. To find the physical location of a record, emulated VSAM creates and maintains an index. This index is updated whenever you add or delete records. The use of indexes is supported through OS/400 logical files.
 - Entry-sequenced (ESDS)
An ESDS has each of its records identified by its displacement from the beginning of the file, that is by its relative byte address (RBA). After a record has been added to a file, its RBA stays constant. New records are added to the end of the file. You cannot delete records, nor alter their lengths.
 - Relative record (RRDS)
An RRDS has fixed-length slots in which records can be stored. Each slot has a unique relative record number (RRN). Each record is therefore identified by its RRN. Records can be inserted or deleted without affecting the position of other data records.

File functions available are reading, writing, updating, and deleting, plus sequential reading, that is, **browsing** in CICS/400 terms, both forwards and backwards through the file.

- Databases
A relational database manager is provided as an integral part of OS/400. It can be accessed through OS/400 file operations or through the IBM DB2/400 Query Manager and SQL Development Kit. SQL statements may be embedded in application program source code. The CICS/400 precompiler automatically invokes the SQL precompiler when EXEC SQL statements are detected within the source code. For information about the COBOL/400 precompiler CL command CRTICSCBL and the ILE C precompiler CL command CRTICSC, see the *CICS for iSeries Application Programming Guide*.

Under CICS/400, distributed relational database support is not available. There is no DL/I support on the OS/400. To access an IMS or DL/I database on a mainframe CICS system, you use either the distributed transaction processing function, or the distributed program link function. These are introduced in “Intersystem communication” on page 13; for details see the *CICS for iSeries Intercommunication* manual.

- Temporary storage

Temporary storage is a mechanism provided by CICS/400 for storing data that must be available to more than one transaction. Data in temporary storage tends to be short-lived, with the emphasis on ease of storage and retrieval. You need not allocate temporary storage until it is required and you keep it only for as long as necessary.

Data is written to temporary storage in the form of a queue of items. These items can be read by any number of application programs in any order, and may be updated. The queue is created when the first item is written to it, and is held in an emulated VSAM file. You have the option of creating the queue in main storage or in auxiliary temporary storage.

See “Temporary storage considerations” on page 84 for information on setting up temporary storage facilities. Details of the characteristics and usage of temporary storage are in the *CICS for iSeries Application Programming Guide*.

- Transient data

Transient data is another storage queue mechanism provided by CICS/400. Data created by one transaction can be held in a transient data queue for use by the same transaction or another. Transient data queues might be used for capturing input data from several workstations and applications. When the number of records in the queue reaches a predefined limit, the transaction that processes those records can be started automatically to update a database or create hardcopy output on a printer, for example. This is known as automatic transaction initiation (ATI).

Data is stored sequentially, and therefore transient data is only suitable for those application programs that need to process data sequentially. Unlike temporary storage, there is no random retrieval or updating of records in transient data queues. Also, records can be read only once.

Transient data queues are known as **destinations**. Details of the different destination types and how to use them are given in “Transient data considerations” on page 85.

- Spooling

The CICS/400 printer spooling facility provides support for writing data to OS/400 print spools, thus allowing an application to be customized to the user’s specific environment for printed output. This facility may be used for output to Intelligent Printer Data Stream™ (IPDS) and other advanced function printers. As an alternative to printing, the OS/400 printer spool facility supports output to diskette.

CICS/400 commands may be used within application programs to support spooling, and OS/400 facilities are used to create the spool files.

- Journaling

OS/400 journal management facility, if used, automatically records all data changes to files. OS/400 can be used for reading and processing journal files; there is no equivalent function within CICS/400. OS/400 also provides the system journaling log. OS/400 journals and journal receivers are an integral part of OS/400 commitment control, which is used by CICS/400 for recovery and syncpoint management. For more details see “Journal control considerations” on page 96.

CICS/400 supports user journaling only through CICS/400 commands. This facility provides automatic journaling for records read from and written to application files. User journal files can be accessed using OS/400 facilities and non-CICS programs.

Journals can be archived either automatically or under operator control.

- Syncpoint control

A syncpoint is a point in an application program when changes to resources are committed. The CICS/400 syncpoint control facility performs commit and backout processing. Support includes the provision of commands for inclusion in the application program to mark explicit syncpoints. Implicit syncpoints occur at normal and abnormal task termination. When a syncpoint occurs, either explicitly or implicitly, the CICS/400 syncpoint control facility issues the corresponding OS/400 commit or rollback operation.

When CICS systems are linked by protected conversations, syncpoint control can extend to the resources on all participating systems. The CICS systems will cooperate to ensure that resources are consistently committed or rolled back, even in the event of most system or conversation failures.

Resource management

To run your system, CICS needs information about your other system resources, including software resources such as programs and data, and hardware resources such as terminals, printers, and telecommunications links. Resource definition is the process by which you tell your CICS/400 system which resources to use, what their properties are, and how to use them. Information for each related type of resource may be thought of as being stored in a table. There is one table for each resource; for example, the file control table holds information about files, the terminal control table holds information about terminal devices, and so on. A complete list and further details can be found in Chapter 4, “Defining resources” on page 41.

Each table is an OS/400 file object and has its own physical and logical files. CICS/400 resource definitions are held as records within the physical files.

CICS/400 uses the information in these tables to control the interactions between your resources: between programs and terminals, or between different CICS systems, for example.

For each resource table definition, there are OS/400 CL commands that allow you to add, change, or display information held in these tables. Alternatively, you can use the OS/400 “Work with” panels to perform these operations on resource definitions.

CICS/400 includes a supplied transaction, CEDA, that provides access to the “Work with” panels, allowing you to maintain resource definitions while in a CICS/400 shell environment. Some resources can be changed and installed while CICS/400 is running, but for others you have to close down and restart the control region, before they take effect.

Resource definitions are loaded into the runtime environment at startup. In addition, you can install or modify certain runtime resource definitions after startup and initialization.

A CICS/400 system administrator is responsible for performing runtime modifications or deletions of runtime resource definitions without requiring the installation of resource definitions. This capability is provided by the INQUIRE, SET, and DISCARD commands, which can be used in application programs, or by using the CICS-supplied transaction CEMT. A brief introduction to CEMT is provided in “Operating interfaces” on page 8.

Another feature of CICS/400 resource definition is the autoinstallation of terminal resources. In a distributed processing system, there is likely to be a large number of terminals. Creating resource definitions for each one would not only be time-consuming but the resulting definitions would take up a large amount of storage space. CICS/400 allows for the creation of model terminal definitions that would apply to a group of terminals, for example all the 3270 display devices. When the user accesses CICS, the definition for the terminal is created by the system from the model. The terminal identifier is generated by the **Device masking** element of the system initialization table parameter **CICS device processing (DEVCTL)**. For more information, see “Managing system initialization resource definitions” on page 239. In addition, an autoinstall control program can be written by the user to customize the selection of terminal identifiers.

Intersystem communication

CICS intersystem communication allows a CICS system to access data and to run programs and transactions on remote CICS systems. In addition, distributed applications can be developed using the standard CICS application programming interface (API). Function may be spread between local CICS systems, remote CICS systems, and non-CICS systems that use advanced program-to-program communication (APPC). Function may be distributed to workstation-based clients. Moving the presentation logic to the workstation allows the development of user-friendly front-end processes that can take advantage of the graphical environment available on the workstation.

The communication facilities offered by CICS/400 are:

- **Function shipping**
Function shipping allows your local transaction to request data from a CICS resource located at another CICS system. CICS/400 automatically routes the request to the required system. You are not aware that the request is being satisfied by a remote system.
- **Transaction routing**
Transaction routing allows a user at any workstation to run any transaction in a remote CICS system. The transaction runs on the remote system but appears to the user as if it is running locally.
- **Distributed program link**
Distributed program link (DPL) allows a local program to link to and run a program on another CICS system. When the remote program ends, control is returned to the initiating program. Distributed program link can be used to access host data that cannot be accessed by function shipping, such as DL/I and DB2 databases.
- **Distributed transaction processing**
Distributed transaction processing (DTP) allows you to have a synchronous conversation between a local program and a remote program. The programs use CICS API commands to allocate and control Advanced Program-to-Program Communication (APPC) conversations. All this is transparent to the end user.
- **Asynchronous transaction processing**
Asynchronous transaction processing allows you to initiate a remote transaction. The real significance of asynchronous transaction processing is that it can be used to balance the workload. You can control when the remote transaction is started by including a time in the parameters of the command that starts the remote transaction. This facility is usually considered as a form of function shipping with the resource being a transaction.

- Server support for workstation-based clients
CICS/400 can act as a server to workstation-based client applications, allowing CICS/400 applications to have front-end graphical user interfaces.

Function shipping, distributed program link, and distributed transaction processing, take advantage of fully protected conversations (sync-level 2), where these are available. In this case, syncpoint control extends to both local and remote CICS resources, which means that the communicating systems cooperate to ensure that all involved recoverable resources are either committed or rolled back at each syncpoint. In the event of system or conversation failure, syncpoint control, together with CICS/400 support, takes the necessary steps to protect resources at the next available opportunity.

Transaction management

CICS/400 includes a number of functions that control the running of your transactions:

- Task control

The CICS/400 task control facility initiates transactions, controls resources at syncpoint and task end, and serializes access to resources.

- Program control

CICS/400 uses:

- OS/400 functions for program loading
- CICS/400 commands for handling transfer of control between programs
- CICS/400 commands for loading and unloading data tables and maps

CICS/400 commands are used for condition handling in CICS/400 application programs. The source code translator adds code to translated programs to trap machine event errors.

- Storage management

The storage management component of CICS/400 provides the support to manage system and user, shared and nonshared, main storage for both internal CICS/400 and user application requirements. User application requirements are supported by the CICS/400 API commands.

Shared CICS resources are used and managed through use of shared user spaces within the control region and the OS/400 data queues.

CICS uses two types of storage:

- Shared storage, which is accessible across many CICS address spaces and contains common data that is needed by CICS service and user application modules.
- Nonshared storage, which is accessible only to an individual CICS address space and contains data that is specific to that address space.

CICS/400 routines create and manage dynamically-acquired main storage areas, track available storage within the appropriate storage environment, and ensure that storage requests are serviced. These CICS/400 routines interface with OS/400 facilities for the creation and release of temporary space objects.

For further details and information on how to set up your storage requirements, see “Storage considerations” on page 67.

- Interval control

CICS/400 interval control is used to extract date and time information for application programs and to delay the starting of a transaction.

Application development

CICS/400 offers a number of facilities to assist the application program developer:

- Application programming interface

The CICS/400 application programming interface (API) includes virtually all the commands and functions that you would expect to have in a CICS mainframe system. The only items missing are those that are not applicable to the OS/400 architecture, such as standard and full function BMS, support for terminals other than 3270 and 5250 devices, acquisition of storage above the 16MB line, and program loading. All other standard CICS API commands are available for CICS/400. The API for CICS/400 is defined in the *CICS Family: API Structure* manual.

- Source language translator

All CICS/400 application programs must be written in either AD/Cycle[®] COBOL/400 or ILE C. The translation process converts the embedded EXEC CICS commands into either COBOL/400 or C statements and program calls to the appropriate CICS/400 support modules. An OS/400 precompiler is used to invoke the translator. The input to this procedure is the COBOL/400 or C code with embedded EXEC CICS commands. After successful translation, the resulting output is compiled by the appropriate language compiler. The translator and compiler are described in more detail in the *CICS for iSeries Application Programming Guide*.

- Map translation

CICS/400 includes a translator for maps created using BMS. An OS/400 CL command is used to create the object versions of your maps ready for use in your programs. A COBOL copybook or C header file is also created.

- CICS/400 system programming commands

CICS/400 includes a special group of commands that are used for monitoring the system, for changing certain parameters with the CICS/400 control region, and for resource administration. These commands, which duplicate many of the functions of the supplied transaction CEMT, are executed from within the application program.

- Application testing

There are four supplied transactions that are useful to application programmers for testing applications:

- The temporary storage browse (CEBR) transaction allows authorized users to work in real time with temporary storage queues. You can also use CEBR to transfer a transient data queue to temporary storage in order to look at its contents, and to transfer the data back to a transient data queue. CEBR is described in the *CICS for iSeries Application Programming Guide*.
- The execution diagnostic facility (CEDF) intercepts EXEC CICS commands at various points within an application program, allowing you to see what is happening. Screens sent by the application program are preserved, so that you can converse with the application program during testing. CEDF is described in the *CICS for iSeries Application Programming Guide*.
- The command-level interpreter (CECI) allows you check the syntax of EXEC CICS commands and to run the commands interactively on a terminal screen. CECI interacts with your test system to allow you to create and delete test data and temporary storage queues, or deliberately to introduce wrong data to test out error logic. CECI is described in the *CICS for iSeries Application Programming Guide*.

- The CICS routing transaction CRTE allows you to pass through to a remote system to verify the creation of temporary storage queues, check connections, and so on.
- Problem determination
 - dump

You can dump runtime storage areas, both within a shell environment and within the control region.

OS/400 dump facilities support the EXEC CICS DUMP TRANSACTION command. COBOL transaction dumps are provided by the COBOL/400 dump format and print program. C transaction dumps are handled by the DMPJOB CL command, which is also used to support the control region dump requested by the CEMT PERFORM SNAP command.

System dumps, including unformatted trace and control block information, are provided as part of the First Failure Data Capture process that results from CICS/400 internal errors.
 - CICS/400 trace

You can trace runtime areas, both within a shell environment and within the control region. If CICS/400 internal trace is active, each execution of an EXEC CICS command results in trace entries being written to an in-memory trace table. Alternatively, these trace table entries are written to an OS/400 user space object. (See the CEMT SET AUXTRACE command, “CEMT INQUIRE | SET AUXTRACE” on page 334.) The trace entries can then be formatted and printed by an offline trace print utility program, initiated by a PRTCICSTRC CL command, supplied within CICS/400. In addition, application programmers can invoke CICS/400 tracing from within their programs.
 - Recovery and restart

CICS/400 resources can be recoverable if recovery information is recorded by the OS/400 commitment control and journaling facilities. Such resources are recovered using standard OS/400 functions.

Portability and migration

CICS/400 includes facilities for:

- Porting CICS application programs from other platforms
- Release-to-release compatibility

Porting application programs

A CICS COBOL or C application program that meets the application programming interface requirements of CICS/400, when ported from another CICS system, will behave as it did on the original system.

The following list summarizes the features of the CICS/400 API that may affect the portability of an application:

- Basic Mapping Support (BMS) functions available with CICS/400 are described in the *CICS for iSeries Application Programming Guide*. See the *CICS Family: API Structure* manual for a definition of the BMS macro options supported.
- BMS macro source input is the only acceptable input for map generation.
- AD/Cycle COBOL/400 and ILE C are the compilers supported for CICS/400 application development. The level of language supported by the COBOL/400 and ILE C compilers means that certain changes could be required to mainframe COBOL and C programs.

- EXEC CICS commands must conform to the API defined for CICS/400 in the *CICS Family: API Structure* manual.
- Support is for the command-level application programming interface only. The macro-level interface is not supported.
- File control emulates the VSAM access methods. BDAM and the DL/I database are not supported.
- CICS/400 does *not* support, and cannot be run within, a System/36™ or System/38™ emulated environment.
- Terminal device support for application programs is provided for:
 - 5250 terminals
 - 3270 terminals and printers, including those supporting double-byte display and printing
 - 3151 ASCII terminals
 - SCS printers
 - Emulation of terminal support for 3270 Model 2 and Model 5 devices
- CICS/400 provides for connectivity with other CICS products by using intersystem communication (ISC) facilities for handling the APPC communication protocol. The data may be held on the iSeries or any remote CICS system.

Further information on porting applications is given in the *CICS for iSeries Application Programming Guide*.

Release-to-release compatibility

Because of the addition of new facilities, objects created under previous releases need to be converted before being used under the current release. In order that you can maintain compatibility between installations at the current release level and installations at previous release levels, CICS/400 allows you to:

- Convert CICS/400 objects created under previous releases to the current release (upward migration).
- Create CICS/400 objects on the current release of CICS/400 to be compatible with the previous release (downward migration).

Previous releases that are supported for migration purposes are the last release of the previous version and the previous release of the current version.

For example, you may have a development installation supporting production installations. You wish to migrate the development installation to the new CICS/400 release but maintain the production installations at a previous release. Resource-definition tables created on the development installation after migration to the new release are not directly transferable to the production installations. You need to use the SAVCICSGRP CL command for downward migration. The reverse is also true: resource definitions in use on the production installations must be converted to the new format (upward migration) after being transferred to the development installation. You do this using the INZCICS CL command or the CONVERT option of the STRCICS CL command.

The release-to-release capability of CICS/400 also allows you to create applications either for the current-release development installation or for distribution to the production installations using a supported previous release. You use the TGTRLS option of the CRTICISC or CRTICISCBL CL command. Applications created on a previous release are compatible with the new release.

Further information about release-to-release compatibility can be found in “Migration” on page 21, Chapter 4, “Defining resources” on page 41, Chapter 8, “Administering the control region” on page 107, and “Using the SAVCICSGRP command” on page 140.

Summary

CICS offers transaction management and system intercommunication facilities with special features to assist in application development. Combined with the easy-to-use operator interface of the OS/400, CICS/400 enhances the transaction processing capability of the OS/400 to offer:

- A migration platform for existing mainframe CICS users
 - Release-to-release compatibility
 - An application management system
 - Integration of iSeries into an OLTP enterprise
 - A partner in distributed processing
 - Facilities for application development
-

Where next?

- Chapter 2, “User-based pricing” on page 19 describes how users are charged for using CICS/400.
- Chapter 3, “Installing CICS/400” on page 21 describes installation, migration, and the installation verification procedure.

If you have your CICS/400 system up and running, go to Part 2, “System administration” on page 39.

Chapter 2. User-based pricing

This Chapter describes how IBM charges you for using CICS/400. From Version 3 onwards, the old method of “processor-based one time charges” is replaced by a method of user-based pricing. OS/400 provides the license management functions for controlling access to software offered under these terms. As a basis for charging, the OS/400 license manager counts **concurrent** rather than **registered** users.

User-based pricing allows you to consolidate your applications and users onto larger machines and gives you the flexibility to defer your major license cost until you are ready to go into production.

What is user-based pricing?

User-based pricing is a pricing strategy that takes the number of concurrent users as its primary price factor, rather than the size of the machines in use. The OS/400 license manager keeps a count of how many concurrent users you have and, if you exceed the limit to which you are entitled, issues a warning that you should increase your limit.

How does the license manager count the number of users?

When CICS/400 uses a software license, each active CICS/400 user is said to hold a “user license”. When a new CICS shell is started, it “requests” a user license; the license manager issues the license and increments a usage counter. When a CICS shell is closed, the license manager “releases” the license and decrements the counter.

When updating the counter, the license manager compares the number of concurrent users with the entitled usage. If this figure is exceeded, the license manager issues a warning message. The license manager sends its messages to the OS/400 message queue QSYSOPR, and CICS/400 issues the messages to the external message queue.

To request a user license, CICS/400 places a call to the OS/400 License Manager. To release the license, CICS/400 places another call to the OS/400 License Manager. CICS/400 releases a license only on request from a CICS shell to which it has previously granted a license.

Whenever the usage limit is exceeded, an informational message is issued.

What is a user?

For the purposes of user-based pricing, a CICS/400 user is one of the following:

- **An interactive instance of the STRCICSUSR command.**

A CICS/400 user license must be available for this command to run interactively. The license is normally released when the command ends. If the license is not released for any reason, ending the associated job always releases the license. When a STRCICSUSR command is rejected, the user receives an explanatory message.

- **A batch instance of the STRCICSUSR command that was initiated by the CEMT SET TERMINAL ACQUIRE command.**

A CICS/400 user license must be available for this command to start successfully. The batch shell started to acquire the terminal requests a user license. The license is released when the batch shell ends.

- **An instance of the intersystem communication (ISC) program, AEGISICC.**

A CICS/400 user license must be available for this program to start successfully. If this program is used as a prestart job, the user license is taken immediately the prestart job begins. The license is released when the batch shell ends.

- **An instance of the CICS/400 dynamic routing program, AEGISRTR.**

A CICS/400 user license must be available for this program to start successfully. If this program is used as a prestart job, the user license is not taken until the first unit of work arrives. The license is released when the program ends or a unit of work arrives for a control region different from the previous one. In the latter case, a new license is requested. If the request fails, the program ends.

What is not a user?

For the purposes of user-based pricing, the following CICS/400 tasks are *not* users:

- Control region job (STRCICS)
- Dynamic install of CICS/400 resources:
 - CEDA INSTALL
 - INSCICSGRP command
- Shells that run in batch and do not have a terminal attached
- Outgoing intersystem communication requests

Chapter 3. Installing CICS/400

This chapter covers the following:

- “Installation”
- “Migration”
- “Installation verification procedure” on page 23
- “The IVP process” on page 24
- “DEFCICSRGN” on page 34

Installation

CICS/400 is installed using the OS/400 licensed-programs installation support, which is described in *Software Installation*, Part 6, Installing Additional Licensed Programs in that book describes the installation process in two tasks, of which you need complete only the first. You should follow the steps described and install CICS/400 and also the CICS/400 Sample Applications.

Installing the CICS/400 licensed program will give you access to a library called QCICS. The library contains the CICS code and all the supplied COBOL copybooks and C header files that you will need to use CICS/400. All the CL commands for CICS/400 are copied from library QCICS to your QSYS library, as part of the installation process. You can find what these commands are using the CMDCICS MENU.

Note that if you already have a CICS/400 Version 5 system but wish to re-install, you should first end all the control regions in the existing system.

Installing the CICS/400 Sample Applications will give you access to a library called QCICSSAMP, which contains:

- A COBOL sample application, used for the IVP (see “Installation verification procedure” on page 23) and also described in the *CICS for iSeries Application Programming Guide*
- A C sample application, described in the *CICS for iSeries Application Programming Guide*
- The DEFCICSRGN sample, described in “DEFCICSRGN” on page 34
- The sample autoinstall control program, described in Chapter 11, “Autoinstall for terminal definitions” on page 305

Migration

For this new version of CICS/400, the record format of the resource-definition tables has been updated, because of the addition of new facilities.

Upward migration

You can have a mixture of Version 5 resource definitions and previous-release resource definitions on CICS/400 Version 5 but you need to migrate the old ones before you can use them. You do this by running the INZCICS CL command, after you have installed the new version of CICS/400.

The INZCICS CL command converts the resource definition tables from all previous releases of CICS/400 to Version 5 formats. Additionally, this command will also be used to remove the unnecessary extension files in QCICSSAMP library (ACCTFI0*, ACCTIX0*, DEFCIC0*, FILEA0*, H0*, QCLSRC0*, QCSRC0*, QDDSSR0*, QLBLSR0*, QMAPSR0*, QCMDSR0*).

CL command defaults

The defaults given in the CL command description are those that are supplied with the iSeries system. You should check that your installation has not made any changes to these command default parameters.

INZCICS

Job: B,I Pgm: B,I REXX: B,I Exec

```
▶▶ INZCICS LIB( *ALL ) library-name ▶▶
```

Function

The Initialize CICS (INZCICS) command causes the specified library on the system to be scanned to locate resource-definition tables from previous releases of CICS/400. Any tables found in the selected library are then converted to the current release of CICS/400.

Required parameters

LIB

Enter the name of the OS/400 library that contains the tables to be converted. Possible values are:

***ALL:** All OS/400 libraries are searched for tables that need to be converted. This may take several hours.

library-name: Specify the name of the OS/400 library that contains the tables.

Downward migration

You cannot use resource definitions created on CICS/400 Version 5 on previous releases except by using one of the following migration techniques:

- The SAVCICSGRP command; see “Using the SAVCICSGRP command” on page 140.
- Maintain resource definition CL commands as source files. You can compile them either before distribution by specifying the previous release on the CRTCLPGM command, or after distribution on each system.

You can migrate downwards from CICS/400 V5R2 to the previous release only. For V5R2, the previous release is considered to be V5R1.

Installation verification procedure

To verify the successful installation of CICS/400, a sample application is provided. This application also gives users who are new to the CICS application programming interface (API) an opportunity to see what a CICS program looks like. The sample application is designed to provide online inquiry and maintenance facilities for a customer credit file in a department store. The application uses files, displays, terminals, and printers. It is described fully in *CICS Application Programming Primer (VS COBOL II)*.

You should read this section sequentially and perform the steps in turn. All the information you need is here, but there are cross-references to other parts of the book where further explanations can be found.

To work through the installation verification procedure (IVP), you require sufficient OS/400 security authority. The user profile must have *PGMR equivalent authority or higher.

Note: If your primary national language is not 2924 (US English), and you do not have CICS available in your primary language, the OS/400 objects for CICS national language support, for example commands and help text, will be placed in library QSYS2924. You must place this library in your system library list, to make sure that CICS functions properly. For further information, refer to Globalization topic in the iSeries Information Center.

Installing the sample code from tape

The source code for this application is supplied on the CICS/400 distribution tape. It can be installed using the Licensed Program installation support on the OS/400. See *Software Installation*.

Library QCICSSAMP contains the following source code for this sample:

1. The application source code, which may be found in the following members of the file QLBSRC:

ACCT00	Display menu
ACCT01	Initial request processing
ACCT02	Update processing
ACCT03	Requests for printing
ACCT04	Error processing
ACCTREC	Layout of account record
ACIXREC	Layout of index record
2. The CICS BMS map source ACCTSET, which is a member in the file QMAPSRC.
3. File QCLSRC, which provides a sample CL program, ACCTRES, for CICS/400 resource definition.
4. File QDDSSRC contains OS/400 DDS definitions for the following two files:

ACCTFIL	The sample key-sequenced data set
ACCTIX	The sample alternate index

The IVP process

The IVP can be run in two separate ways:

1. If you have the COBOL/400 compiler installed, you can verify that the application development part of CICS/400 is working properly, and then run the sample transaction ACCT.
2. Alternatively, if you do not have the COBOL/400 compiler, you can choose just to run the application itself. IBM has provided all the necessary COBOL object programs for the application in the CICS/400 sample library and you can omit steps 3 and 4 described below.

As an alternative to the IVP, you could run the sample control region definition program, DEFICSRGN, which is described in "DEFICSRGN" on page 34.

The IVP comprises the following steps:

- "Step 1. Creating a working copy of QCICSSAMP"
- "Step 2. Adding libraries to the OS/400 library list"
- "Step 3. Translating the BMS map" on page 25
- "Step 4. Translating the program" on page 26
- "Step 5. Creating the recoverable/nonrecoverable resources" on page 27
- "Step 6. Journaling the CICS/400 recoverable resources" on page 28
- "Step 7. Defining the CICS/400 resources" on page 28
- "Step 8. Creating the CL program" on page 28
- "Step 9. Starting the CICS/400 control region" on page 33
- "Step 10. Starting the sample transaction" on page 33

In generating the BMS and COBOL objects, you will use the supplied CL commands CRTICSMAP (to create the symbolic and physical BMS maps) and CRTICSCBL (to translate and compile COBOL source code containing CICS API into OS/400 program objects).

Step 1. Creating a working copy of QCICSSAMP

Before generating the BMS and COBOL objects, you should make a security copy of the QCICSSAMP library to CICSWORK by issuing the following command:

```
COPYLIB FROMLIB(QCICSSAMP) TOLIB(CICSWORK)
```

When the copy is complete, you will see the message:

```
Library QCICSSAMP copied to library CICSWORK
```

This leaves QCICSSAMP as a backup in case of errors. CICSWORK is your working copy and is referred to in the rest of this chapter.

Note: If you choose to use a different library name, all references to CICSWORK in the following sections should be replaced by your own library name. You will need access to an editor to do this in "Step 7. Defining the CICS/400 resources" on page 28.

Step 2. Adding libraries to the OS/400 library list

To use some of the OS/400 CL commands, you need to add the CICSWORK library to the OS/400 library list. You do this by issuing the following command:

```
ADDLIBLE CICSWORK
```

When the library has been added, you will see the message:

```
Library CICSWORK added to the library list
```

Before you attempt to compile the sample program, you should add the QCICS library to the OS/400 library list. File QLBSRC in this library holds four copybooks necessary for the compilation of the sample program. You add this library by issuing the following command:

```
ADDLIBLE QCICS
```

When the library has been added, you will see the message:

```
Library QCICS added to the library list
```

The DSPLIBL CL command shows the entire library list. The EDTLIBL CL command shows the user part of the library list and allows you to change items in the library list.

Step 3. Translating the BMS map

Basic Mapping Support (BMS) is the component of CICS/400 that handles the display, mapping, and receiving of data at display terminals. The layout of data is described to BMS in objects known as maps. A map consists of:

- A source definition
- A translated program object (physical map) used by BMS
- A COBOL/400 copybook (symbolic map) used by a COBOL application program

Translating the BMS map using the CRTICSMAP CL command produces physical and symbolic map objects from the source definition. These objects are used during execution of the sample program. Translation also produces a COBOL/400 copybook, which is later copied into the COBOL programs during compilation.

For this sample program, CRTICSMAP translates the source member ACCTSET of the file QMAPSRC to the physical map object ACCTSET and the symbolic map member ACCTSET of file QLBSRC. If the copybook but not the source file is specified in the COPY statement of the COBOL program, the COBOL compiler assumes that the copybook source is in QLBSRC. Therefore it is recommended that all copybooks are held in QLBSRC.

To translate the map, enter:

```
CRTICSMAP
```

on the CICS/400 command line. Press PF10 to display the additional parameters. The panel shown in Figure 2 on page 26 is displayed. You need type only in the four fields indicated. The remaining fields have acceptable defaults.

```

Create CICS Map (CRTCICSMAP)

Type choices, press Enter.
Map object name . . . . . Name
Library . . . . . *CURLIB Name, *CURLIB
Map source file . . . . . QMAPSRC Name
Library . . . . . *LIBL Name, *LIBL, *CURLIB
Map source member name . . . . *MAP Name, *MAP

Additional Parameters

Symbolic map source file . . . *DEFAULT Name, *DEFAULT
Library . . . . . *CURLIB Name, *LIBL, *CURLIB
Symbolic map source member . . *MAP Name, *MAP
Replace output objects . . . *YES *YES, *NO
Target release . . . . . *CURRENT *CURRENT, *PRV, V5R2M0...
Text . . . . . *SRCMBRTXT

Bottom

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

```

Figure 2. The Create CICS map screen

Notes:

- 1** Enter ACCTSET. This is the name of the physical map that is generated for later use when the sample transaction is executed. The object that is generated is an OS/400 user space object of the same name as specified here.
- 2** Enter CICSWORK. This is the name for the library where the physical map is to be created.
- 3** Enter CICSWORK. This is the name of the library where the map source is located.
- 4** Enter CICSWORK. This is the name of the library where the QLBSRC file COBOL copybook is located.

When you have completed the panel, press Enter.

Step 4. Translating the program

Translate and compile the COBOL programs (ACCT00, ACCT01, ACCT02, ACCT03, and ACCT04) using the CICS/400 supplied CL command CRTCICSCBL. To translate the program, enter:
CRTCICSCBL

The panel in Figure 3 on page 27 is displayed. You need type only in the three fields indicated. The remaining fields have acceptable defaults.


```

                                Create CICS COBOL (CRTCICSCBL)

Type choices, press Enter.

Program . . . . . PGM          > ACCT00      1
Library . . . . .           > CICSWORK     2
Source File . . . . . SRCFILE  QLBSRC
Library . . . . .           > CICSWORK     3
Source Member . . . . . SRCMBR  *PGM
Commitment control . . . . . COMMIT *CHG
Text description . . . . . TEXT  *SRCMBRTXT

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

```

Figure 3. The Create CICS COBOL screen

Notes:

- 1 This is the name of the OS/400 program object that is generated for later use when the sample transaction is executed.
- 2 This is the library that holds the OS/400 program object.
- 3 This is the name of the library that holds the program source code.

Program objects for each of the sample programs are supplied in the library CICSWORK, but you may recompile them all if you wish. To compile the other programs, repeat the CRTCICSCBL command, making the same entries in the same three highlighted fields, changing the **Program** field as necessary to ACCT01, ACCT02, and so on.

Further details of the CRTCICSCBL and CRTCICSMAP CL commands can be found in *CICS for iSeries Application Programming Guide*, but you do not need to refer to them at this stage.

Step 5. Creating the recoverable/nonrecoverable resources

In the QCICS library there are two sample files: AAEGCICSTR, which is used for recoverable temporary storage and transient data queues, and AAEGCICSTN, which is used for nonrecoverable temporary storage and transient data queues. These files need to be copied from the QCICS library to the CICSWORK library. Use the OS/400 CL command CRTDUPOBJ to copy them, noting that the four characters following AAEG are the four-character CICS/400 system ID. For this IVP, use SAMP as the system ID. Enter the following commands:

```
CRTDUPOBJ OBJ(AAEGCICSTR) FROMLIB(QCICS) OBJTYPE(*FILE)
          TOLIB(CICSWORK) NEWOBJ(AAEGSAMPTR)
```

```
CRTDUPOBJ OBJ(AAEGCICSTN) FROMLIB(QCICS) OBJTYPE(*FILE)
          TOLIB(CICSWORK) NEWOBJ(AAEGSAMPTN)
```

Step 6. Journaling the CICS/400 recoverable resources

Set up an OS/400 journal file for the previously-created recoverable file AAEGSAMPTR. To do this, enter the following commands:

```
CRTJRNRCV JRNRCV(CICSWORK/SAMPJRNRCV)
          TEXT('Journal receiver for CICS/400 system id SAMP')
```

```
CRTJRN JRN(CICSWORK/SAMPJRN) JRNRCV(CICSWORK/SAMPJRNRCV)
        TEXT('Journal for CICS/400 system identifier SAMP')
```

Start journaling the recoverable physical file by entering:

```
STRJRNPF FILE(CICSWORK/AAEGSAMPTR) JRN(CICSWORK/SAMPJRN)
          IMAGES(*BOTH)
```

You will see the message

```
1  FILES HAVE STARTED JOURNALING
```

For more information on journals, see “Recovery considerations” on page 104.

Step 7. Defining the CICS/400 resources

In most cases, you would not use a program to define resources. You would use a CL command or a menu-driven approach using the CEDA transaction. See “Using CEDA resource definition online” on page 58. For the IVP, a program to define resources has been supplied, to speed up the resource definition stage.

This program is in member ACCTRES in file QCLSRC in library CICSWORK. To run the IVP you do not need to make any changes to it but the source is provided in case you wish to make any changes to it later.

Step 8. Creating the CL program

To create the OS/400 CL program from this source you use the OS/400 CL command CRTCLPGM. Further details on the parameters for this command can be found in *CL Programming*. Alternatively, if you are interested only in creating the object, you can enter:

```
CRTCLPGM PGM(CICSWORK/ACCTRES) SRCFILE(CICSWORK/QCLSRC)
          SRCMBR(ACCTRES)
```

When this program has been compiled, run it by entering the following:

```
CALL CICSWORK/ACCTRES CICSWORK
```

where *CICSWORK* is the name of the target library.

This program runs for several minutes (depending on size and loading of your OS/400) and creates all the resource definition objects required by the sample program in the named library, using the details supplied. When this has finished successfully, you can start up the CICS/400 control region.

Figure 4 on page 29 is an annotated listing of the CL source program ACCTRES, with indicators showing where the code may need to be changed and why.

```

/*****/
/*
/* MODULE NAME = ACCTRES
/*
/* DESCRIPTIVE NAME = SAMPLE OS/400 CL PROGRAM USED TO DEFINE
/* RESOURCES FOR CICS/400 SAMPLE PROGRAMS IN
/* BATCH.
/*
/* 5763-DFH
/*
/* INPUTS: LIB - OS/400 LIBRARY THAT IS BEING USED FOR THE
/* SUPPLIED SAMPLE APPLICATION. THIS PARAMETER IS NORMALLY
/* PASSED TO THIS PROGRAM BY THE CL PROGRAM CRTSAMP, BUT IF
/* REQUIRED YOU CAN RUN THIS PROGRAM INDEPENDENTLY FROM
/* CRTSAMP, SUPPLYING YOUR OWN LIBRARY NAME.
/*
/* STATUS = CICS/400 VERSION 5
/*
/*****/

/*****/
/* START THE PROGRAM AND DEFINE THE PASSED PARAMETER.
/*
/*****/
PGM PARM(&LIB)
DCL VAR(&LIB) TYPE(*CHAR) LEN(10)

/*****/
/* CREATE THE CICS/400 GROUP TO HOLD THESE DEFINITIONS.
/*
/*****/
1 CRTCICSGRP LIB(&LIB) GROUP(ACCT) TEXT('CICS/400 +
sample transactions group.') +
RECOVER(*YES) CLRGROUP(*YES)

/*****/
/* FIRSTLY WE DEFINE THE CICS SYSTEM INITIALIZATION TABLE ENTRY.
/*
/*****/
2 ADDCICSSIT LIB(&LIB) GROUP(ACCT) GLTLIB(&LIB) +
GLTGRP(ACCT) WRKARASIZE(100) DUMP(*NO) +
DEVCTL(3457) INTRCCTL(7000 *YES *NO) +
AUXTRCCTL(*YES *NO &LIB/SAMPTRACE1 +
&LIB/SAMPTRACE2) USRTRC(*YES) TSCTL(0 +
*NO) SHRSTG((400 20 10) (400 20 10) (400 +
20 10)) NONSHRSTG((200 20 10) (200 20 10) +
(200 20 10)) TDCTL(*INVOKER *NO)

```

Figure 4. Resource definitions for the IVP (Part 1 of 4)

IVP

```
/* *****  
/* NEXT WE DEFINE TWO DEVICES TO BE USED BY THE TRANSACTIONS IN THIS */  
/* SAMPLE. */  
/* NOTE: ALTHOUGH THE 'LIB' PARAMETER CAN STAY THE SAME YOU SHOULD */  
/* CHANGE THE 'DEVD' PARAMETER TO MATCH ONE ON YOUR SYSTEM. YOU MAY */  
/* ALSO CHOOSE TO ADD ADDITIONAL TERMINAL DEFINITIONS IF YOU SO */  
/* WISH. */  
/* *****  
3 ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(TM01) +  
      DEVD(DSP01) DEVMODEL(*BOTH) ATISTS(*YES) +  
      UCTRN(*YES) ALTSCN(24X80) SHIP(*YES) +  
      DEVACQ(*YES)  
  
      ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(TM02) +  
      DEVTYPE(3270) DEVD(DSP02) DEVMODEL(*BOTH) +  
      ATISTS(*YES) UCTRN(*YES) ALTSCN(24X80) +  
      SHIP(*YES) DEVACQ(*YES)  
  
      ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(TM03) +  
      DEVTYPE(3270J) DEVD(DSP03) +  
      DEVMODEL(*BOTH) ATISTS(*YES) UCTRN(*YES) +  
      ALTSCN(24X80) SHIP(*YES) DEVACQ(*YES)  
  
      ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(TM04) +  
      DEVTYPE(3151) DEVD(DSP04) DEVMODEL(*BOTH) +  
      ATISTS(*YES) UCTRN(*YES) ALTSCN(24X80) +  
      SHIP(*YES) DEVACQ(*YES)  
  
      ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(PR01) +  
      DEVTYPE(SCS) PRTFILE(QSYSVRT) DEVD(PRT01) +  
      UCTRN(*YES) SHIP(*YES) DEVACQ(*YES)  
  
      ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(PR02) +  
      DEVTYPE(3270JP) PRTFILE(QSYSVRT) +  
      DEVD(PRT02) UCTRN(*YES) SHIP(*YES) +  
      DEVACQ(*YES)  
  
      ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(PR03) +  
      DEVTYPE(3270P) DEVD(PRT03) UCTRN(*YES) +  
      SHIP(*YES) DEVACQ(*YES)  
      ADDCICSTCT LIB(&LIB) GROUP(ACCT) CICSDEV(L860) +  
      DEVTYPE(SCS) DEVD(PRT01) UCTRN(*YES) +  
      SHIP(*YES) DEVACQ(*YES)  
  
/* *****  
/* THE SECTION BELOW DEFINES THE TRANSACTION SPECIFIC RESOURCES */  
/* NEEDED TO RUN THE SAMPLE APPLICATIONS. IF YOU CHANGE THEM THEN */  
/* THE SAMPLES MAY NO LONGER RUN. */  
/* *****  
4 ADDCICSFCT LIB(&LIB) GROUP(ACCT) FILEID(ACCTFIL) +  
      FILE(&LIB/ACCTFIL) MBR(ACCTFIL) +  
      RECOVER(*NO) RCDACT(*ADD *NOBROWSE *DLT +  
      *READ *UPD)  
      ADDCICSFCT LIB(&LIB) GROUP(ACCT) FILEID(ACCTIX) +  
      FILE(&LIB/ACCTIX) MBR(ACCTIX) +  
      RECOVER(*NO) RCDACT(*ADD *BROWSE *DLT +  
      *READ *UPD)
```

Figure 4. Resource definitions for the IVP (Part 2 of 4)

```

/*****
/* NOW WE DEFINE THE GROUPS OF DEFINITIONS THAT WE WISH TO USE FOR */
/* THIS CONTROL REGION. IN THIS EXAMPLE WE ARE INSTALLING JUST THOSE */
/* USED FOR THIS SET OF SAMPLE TRANSACTIONS AS WELL AS THE CICS */
/* SUPPLIED TRANSACTIONS. */
/*****
5      ADDCICSGLT LIB(&LIB) GROUP(ACCT) INSLIB(&LIB) +
          INSGRP(ACCT)

          ADDCICSGLT LIB(&LIB) GROUP(ACCT) INSLIB(QCICS) +
          INSGRP(AEGEDF)

          ADDCICSGLT LIB(&LIB) GROUP(ACCT) INSLIB(QCICS) +
          INSGRP(AEGINTER)

          ADDCICSGLT LIB(&LIB) GROUP(ACCT) INSLIB(QCICS) +
          INSGRP(AEGOPER)

          ADDCICSGLT LIB(&LIB) GROUP(ACCT) INSLIB(QCICS) +
          INSGRP(AEGSPI)

/*****
/* DEFINE THE PROGRAMS THAT ARE USED WHEN EACH TRANSACTION THAT IS */
/* SUPPLIED IN THIS EXAMPLE IS EXECUTED. */
/*****
6      ADDCICSPT LIB(&LIB) GROUP(ACCT) PGMID(ACCT00) +
          PGMOBJ(&LIB/ACCT00)

          ADDCICSPT LIB(&LIB) GROUP(ACCT) PGMID(ACCT01) +
          PGMOBJ(&LIB/ACCT01)

          ADDCICSPT LIB(&LIB) GROUP(ACCT) PGMID(ACCT02) +
          PGMOBJ(&LIB/ACCT02)

          ADDCICSPT LIB(&LIB) GROUP(ACCT) PGMID(ACCT03) +
          PGMOBJ(&LIB/ACCT03)

          ADDCICSPT LIB(&LIB) GROUP(ACCT) PGMID(ACCT04) +
          PGMOBJ(&LIB/ACCT04)

          ADDCICSPT LIB(&LIB) GROUP(ACCT) PGMID(ACCTSET) +
          CICSMAP(*YES) PGMOBJ(&LIB/ACCTSET)

```

Figure 4. Resource definitions for the IVP (Part 3 of 4)

```

/*****
/* DEFINE THE TRANSACTIONS THAT ARE USED IN THIS SAMPLE.          */
/*****
7      ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(ACCT) +
          PGMID(ACCT00)

          ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(ACEL) +
          PGMID(ACCT03)

          ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(ACLG) +
          PGMID(ACCT03)

          ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(AC01) +
          PGMID(ACCT01)

          ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(AC02) +
          PGMID(ACCT02)

          ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(AC03) +
          PGMID(ACCT03)

          ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(AC05) +
          PGMID(ACCT03)

          ADDCICSPCT LIB(&LIB) GROUP(ACCT) TRANSID(AC06) +
          PGMID(ACCT03)
/*****
/* ADD A DCT ENTRY FOR A CSMT LOG FILE.                            */
/*****
8      ADDCICSDCT LIB(&LIB) GROUP(ACCT) DEST(CSMT) +
          TYPE(*INTERNAL)

          ENDPGM

```

Figure 4. Resource definitions for the IVP (Part 4 of 4)

Notes:The following notes apply to the above source:

- 1** The CRTICISGRP CL command is used here to create a group to hold the resource definitions that will be used when the control region is started.

Note: RECOVER(*YES) and CLRGROUP(*YES) are specified. This means that any previous resource definitions done for this group will be lost when this program is run.
- 2** The system initialization table entries contain various parameters that are used when the control region is started. In this example, some of the parameters have been changed from their defaults, to illustrate some of the things that you can do.
- 3** The ADDCICSTCT command is used to define devices to CICS. If you want to add the capability for more terminals and printers to run against this control region, you will have to add additional ADDCICSTCT commands.
- 4** The ADDCICSFCT command adds file control table entries for the two file objects that will be used by the sample transaction. If you specified a different library from CICSWORK when you created these OS/400 physical file objects earlier, do not forget to change it here as well.
- 5** When this control region is started, all the IBM supplied transactions are required. More details of these transactions can be found in Part 4, "CICS-supplied transactions" on page 321.

- 6** The ADDCICSPPT command is used to define programs that are used by the sample transaction. If you worked through steps 2 and 3 of the IVP to regenerate the COBOL/400 object code from the supplied source, make sure that you specify the correct OS/400 library name.

Within the processing program tables (PPT) you also specify the location of the BMS objects. The CICS MAP(*YES) operand is specified to identify that the object is a CICS BMS map rather than a program object.

- 7** The transactions that can be run under this control region are defined.
- 8** Finally, an intrapartition transient data queue for the CSMT log is defined.

Step 9. Starting the CICS/400 control region

In this example, the control region is run as a batch OS/400 job. To do this enter the following:

```
SBMJOB CMD(STRCICS CTLRGN(SAMP) STRTYPE(*COLD) SITLIB(CICSWORK)
        SITGRP(ACCT))
```

where SAMP is the four-character system ID that you specified earlier when you created the TD and TS files.

This command submits an OS/400 job and you must make sure that any additional parameters that you may want to specify, like job description, are added to the command invocation string. Details of the additional parameters and what they do can be found in *CL Programming*.

When the job has started, you watch its progress by using the OS/400 CL command

```
WRKACTJOB
```

Pressing PF5 while this screen is active refreshes its contents. When the status of the control region job has changed to DEQW, you can start the sample transactions. For details of other statuses, refer to *Work Management*.

Step 10. Starting the sample transaction

To start the sample transaction, enter the following from an OS/400 CL command line:

```
STRCICSUSR CTLRGN(SAMP) TRANID(ACCT)
```

where SAMP is the four-character CICS/400 system identifier specified earlier.

The terminal that you use to do this should be one that you specified earlier in the TCT entries. (The DEVD parameter of the ADDCICSTCT CL command identifies the name of the OS/400 device that can be used.) If the terminal from which you issue the command has not been specified in the TCT entry, there is a small delay while CICS creates the necessary internal definitions for the terminal to work with this transaction.

You will now see the first data entry screen of the sample transaction.

The appearance of this screen verifies that the IVP has completed successfully. You do not need to do anything other than press CLEAR to exit. However, ACCT provides a simple update application that enables you to add, update, and delete records to verify further that your system has been installed correctly. If you would like to try out some of these actions, see the *CICS for iSeries Application*

Function

The Define CICS Region (DEFCICSRGN) CL command creates automatically resource definitions for your CICS/400 system and starts a user shell.

Required parameters**Region name (CTLRGN)**

Enter the four-character system identifier of the CICS/400 control region that you wish to create.

Library to create to hold data (LIB)

Enter the name of the target library that will hold the CICS/400 definitions and the OS/400 objects pertaining to the control region. This library must not already exist.

Name of CICS group to hold SIT (SITGRP)

Enter the name of the top-level group to hold the system initialization table (SIT) and the group list table (GLT) for the control region.

Optional parameters**LIBLIST**

Enter the names of libraries, up to a maximum of 30, that already contain the resources you wish to have defined. For each library in the list, autodefinition create a CICS group containing definitions for the CICS/400 programs, maps and data files within that library, that meet the resource definition creation criteria.

Running DEFCICSRGN

Running DEFCICSRGN against QCICSSAMP would give you a good idea of what the autodefinition code can do. Your userid would need *ALLOBJ authority.

First, you must add library QCICSSAMP to your library list using the command:

```
ADDLIBLE QCICSSAMP
```

Then type

```
DEFCICSRGN
```

and enter F4 for prompts. You will be prompted for the control region name, library name, group name, and library list.

Then you will be presented with a panel listing all the possible resources, and asking you to enter how many of each you expect to have in your system. DEFCICSRGN uses this information to calculate how much storage it should allocate. Refer to Chapter 5, "Defining a basic control region" on page 61

When you have answered the storage question, you are taken into a CICS/400 user shell, and will see the following screen:

```

Welcome to CICS for OS/400 V5, region FRED, on 05/07/02 15:27:16

CCCCCCC IIIIIIII CCCCCC  SSSSS // 444 00000 00000
CCCCCCCC IIIIIIII CCCCCCCC SSSSSSS // 4444 0000000 0000000
CC CC II CC CC SS SS // 44444 00 0000 00 0000
CC II CC SSS // 44 44 00 00 00 00 00 00
CC II CC SSS // 44444444 00 00 00 00 00 00
CC CC II CC CC SS SS // 444444444 0000 00 0000 00
CCCCCCCC IIIIIIII CCCCCCCC SSSSSSS // 44 0000000 0000000
CCCCCCC IIIIIIII CCCCCC SSSSS // 44 00000 00000

=====
APPLID FRED LCLNETID GBIBMIYA SYSNAME WINAS15
JOBNUM 002665 USERNAME CLAYTON JOBNAME QPADEV0002
=====

Press ENTER to continue

```

Figure 5. Welcome to CICS/400 screen

Your CICS system is now up and running. You may clear the screen, and enter a transaction code, for example ACCT.

Autodefined resources

BMS maps

CICS/400 BMS maps are stored in user space objects with an object attribute of **CICS400BMS**. Within CICS/400, BMS maps are identified uniquely by a program id of up to eight characters. Because of this, the autodefinition program adds definitions only for maps with a maximum of eight characters in their object names.

CICS/400 programs

All program objects with a maximum of eight characters in their object names, and with an object attribute of **CBL** or **CLE**, are defined as CICS/400 resources.

In addition, transactions are defined in the following cases:

- If the program name is four characters or less, a transaction will be defined under that name.
- If the text description of the object begins with **Trans**, a transaction will be defined for each word which follows **Trans**, for example, ACCT03 in QCICSSAMP. This allows a single program to be referenced by more than one transaction.

OS/400 data file members

Within CICS/400, files are identified uniquely by a file identifier of up to eight characters. Each CICS/400 file relates to a member within an OS/400 data file. Autodefinition looks at the members within the files in the given list of libraries. For each member with eight characters or less in its member name, autodefinition builds a CICS/400 file definition under the same name. In addition:

- OS/400 source files are ignored
- Files using a shared access path are ignored

Member names should be unique across all files within all the libraries that are to be accessed by autodefinition in order to avoid naming conflicts.

The autodefinition program attempts to work out how to define the file members to CICS/400 by looking at many of the OS/400 file attributes. Table 1 lists the file attributes that are inspected and how they affect the definition that autodefinition creates.

Table 1. File attributes inspected during autodefinition

OS/400 file or member attributes	Effect on autodefinition
OS/400 file is being journaled.	Members are defined to CICS/400 as being recoverable.
File field definition includes a field which is variable length.	Members are defined as having a variable record length.
File has keyed access.	Members are defined as key-sequenced files (KSDS).
File has entry-sequenced access and variable length records.	Members are defined as entry-sequenced files (ESDS).
File has entry sequence access and fixed length records. Members have some records deleted.	Relative record files are emulated on CICS/400 by initializing the member with deleted records. The presence of deleted records is therefore taken to imply that the file should be defined as a relative record sequenced file. This is a "best guess".
File has entry-sequence access and fixed-length records. Members have no records deleted.	Members are defined as entry-sequenced files (ESDS).

For further information on CICS/400 files, see "File control considerations" on page 88.

Other definitions

In addition to these resource definitions, the following objects and resources that will be used later by the CICS/400 control region are defined:

- Files to hold recoverable and nonrecoverable temporary storage (TS) and transient data (TD) queues.
- Journal receiver and journal objects for journaling the file which holds the recoverable TS and TD queues.
- An explicit terminal definition for the terminal running the autodefinition program.
- Model terminal definitions for 3270 and 5250 devices.
- A default printer.
- Many of the IBM supplied transactions.
- A subsystem description for a subsystem in which the CICS/400 control region will run.
- A job queue and class description for use by the subsystem.
- A configuration list entry in the local configuration list, if there is one (for APPN).

Storage

When all these definitions have been created, autodefinition uses the sample command CHGCICSSIZ, which presents you with a list of questions on how

the control region is to be used and what sort of transactions are to be run in it. This is essentially the same information as that listed in Table 11 on page 70.

Using this information, and a count of all the CICS/400 resource definition table entries, a suitable set of storage parameters for the SIT are calculated and applied.

Starting the control region

Finally, autodefinition submits a batch job to bring up the control region. It uses the command STRCICSDLY (Start CICS Delay), which is one of the autodefinition sample programs, to start the appropriate subsystem and control region in an orderly fashion, and it waits for the control region to become available. You may use the STRCICSDLY command in the future to bring up a subsystem and control region that were defined using the DEFCICSRGN command.

When you have seen what DEFCICSRGN does, you may wish to customize parts of the code, in order to minimize the amount of tailoring you might need to do to the definitions. See the README member in DEFCICSSRC for details.

Deleting the autodefined control region

If you wish to delete the CICS/400 control region that has been created:

- End the CICS system, using the ENDCICS CL command.
- End the subsystem in which it is running. The subsystem has the same name as the control region.
- Delete the library that was specified in the CICSLIB parameter of the DEFCICSRGN command.
- Remove the configuration list entry from the local configuration list, if applicable.

Part 2. System administration

Chapter 4. Defining resources	41	OS/400 command security	76
What is resource definition?	41	Limited capability checking	76
Gathering preliminary information	41	Resource checking of command program objects	76
Resource definition tables	42	Resource security	77
Resource definition architecture.	42	Supplied transactions	77
Creating resource definitions	45	User transactions	77
Defining and maintaining group definitions	45	Application example	77
What is a resource definition group?	46	Adoptive authority	78
Using the resource definition CL commands	46	Communication security	78
Specifying which groups are used	47	User profile and group profile security	78
Recovering groups	48		
Installing a group	48	Chapter 7. Defining your own control region	81
Using the ADDCICSGLT command	48	OS/400 data management considerations	82
Using the INSCICSGRP command.	48	Transaction, program, and map considerations.	82
Using the CEDA Install group function	49	Example PCT definitions	83
Defining and maintaining individual resource definitions.	51	Example PPT definitions	83
How to use the resource definition commands.	52	Temporary storage considerations	84
Entering CL commands from the command line	52	Recoverable and nonrecoverable TS queues.	84
Using the prompt screens.	52	Example TST definitions	85
Displaying a prompt screen	52	Transient data considerations	85
Displaying parameter keywords	53	Extrapartition transient data.	86
Entering resource and transaction identifiers	54	Recoverable intrapartition TD files.	86
Getting further parameters	54	Example DCT definitions.	86
Getting help	54	Defining a CSMT log destination	87
Using WRKCICSxxx commands	54	File control considerations	88
Working with CICS groups	54	Supported file types	88
Working with tables	56	Automatic file closure	89
Working with resource definitions	56	When does automatic file closure occur?.	90
Using CEDA resource definition online	58	Shared open data paths restrictions	90
		Recoverable files.	90
		Example FCT definitions	90
		KSDS examples	91
		ESDS examples	92
		RRDS example	93
Chapter 5. Defining a basic control region	61	Interval control considerations	94
Defining the system initialization table (SIT)	61	Timer-related tasks	94
Defining temporary storage and transient data files	63	Protected interval control start requests	94
File characteristics	63	Interval control elements	95
Creating the physical files	63	Acquiring terminals	95
Defining supplied transactions	65	Interval control batch shells	95
Setting default wait times.	67	Request identifiers	96
Setting job priorities	67	Start requests and autoinstalled terminals	96
Storage considerations.	67	Journal control considerations	96
Control region storage objects	68	Journal records	96
Estimating control region storage requirements	68	Journal output synchronization.	97
The control region AEGSAMPLSO space object	69	Creating CICS/400 user journals	97
The control region AEGSAMPUSYS space object	69	Journal prefix area	97
The control region AEGSAMPUSR space object	71	User journal restriction	97
Shell storage objects	72	Using non-switchable journals	97
The shell AEGSAMPLSO space object	73	Using switchable journals	97
The shell AEGSAMPUSYS space object.	73	Example creation of CICS/400 journal	99
The shell AEGSAMPUSR space object	74	Example JCT definitions.	100
Summary	74	Device considerations	100
		Defining local CICS terminals	100
Chapter 6. Security requirements for CICS/400	75	Example TCT definitions	100
Signon security and display station security	75	Defining remote systems	101
Initial menu and initial program security	76		

Example TCS entry	101
Remote resource considerations	102
Defining remote resources	102
Defining remote terminals	102
Example TCT definitions for remote and shippable terminals	102
Printer spooling considerations	102
Trace considerations	104
Internal trace	104
Printing CICS/400 trace	104
Recovery considerations	104
Recoverable file commitment control requirements	104
OS/400 commitment control and logical unit of work (LUW).	105
Chapter 8. Administering the control region	107
Introduction to the control region	107
Control region processing	108
Control region initialization	108
Control region runtime processing	109
Handling processing requests	110
Communication between control region and shell	110
Control region objects	110
Control region shutdown	111
Starting a control region (STRCICS)	112
STRCICS	112
Setting the STRTYPE parameter	114
Terminology	114
Operation with no in-doubt units of work	115
Operation with in-doubt units of work	115
CICS/400 recovery with a cold start	116
CICS/400 recovery with a warm start	116
CICS/400 recovery with an emergency start	117
Ending a control region (ENDCICS)	118
ENDCICS	118
Chapter 9. Administering a CICS/400 shell.	121
Introduction to the CICS/400 shell	121
Shell objects	121
Shell processing	122
Shell initialization	122
Runtime processing	123
Shell shutdown.	124
Starting a user shell (STRCICSUSR)	125
STRCICSUSR	125
Ending a user shell (ENDCICSUSR)	127
ENDCICSUSR	128
CESF—CICS sign-off transaction	129

Chapter 4. Defining resources

Chapter 1, “Introducing CICS® for iSeries™” on page 3 introduced the concept of resource definition and Chapter 3, “Installing CICS/400” on page 21 gives the resource definitions required in order to run the IVP. This chapter describes the resource definition tables used by CICS/400 and how to use the resource definition commands that are described in Chapter 10, “Defining resources-reference information” on page 133. If you are new to CICS, you will need the information in this chapter before you read Chapter 5, “Defining a basic control region” on page 61 and Chapter 7, “Defining your own control region” on page 81.

The material in this chapter is supplemented by Chapter 10, “Defining resources-reference information” on page 133, which contains syntax and parameter descriptions for each resource definition CL command.

This chapter covers the following:

- “What is resource definition?”
- “Resource definition architecture” on page 42
- “Creating resource definitions” on page 45
- “Defining and maintaining group definitions” on page 45
- “Installing a group” on page 48
- “Defining and maintaining individual resource definitions” on page 51
- “How to use the resource definition commands” on page 52

What is resource definition?

Resource definition is the process by which you tell your system which resources to use, what their properties are, and how CICS can use them. CICS/400 must know the configuration of terminals and logical units in the network. To process transactions, it must be able to load and execute the appropriate program for each expected transaction type. To recover data in the event of a problem with the system, it must be able to locate temporary storage queues.

Gathering preliminary information

Before you start, you need to gather all the information necessary to define the resources to the system, based on business requirements, information provided by either the application developer or the packaged software application. In addition, you have to create the underlying OS/400 objects required by the applications.

To understand the requirements, you need the answers to a number of questions. For instance:

- What is the system ID (SYSID) or control region name (CTRLGN) of the system you are setting up?
- Which functions and transactions is each user going to use on the system?
- Will the system communicate with other systems?
- Does the system require directly-connected printers?
- What applications are going to be run on the system?
- Where are the program objects and the map sets going to be located?
- Which files do the applications use?

- Are the files recoverable?
- How does the application use those files?
- What kind of access is needed for the files? Keyed-sequence, random, or sequential?
- Do the applications require transient data, temporary storage, or both?
- Does the system need to use any of the supplied transactions?
- What are the names of the resources and their corresponding OS/400 objects?

Resource definition tables

Resources are stored in resource definition tables. CICS/400 uses the information in the tables to control the interactions between, for example, programs and terminals, transaction identifiers and programs, between different CICS systems, and so on.

There are 11 different resource definition tables: some, like the system initialization table (SIT), need to be defined before CICS can be executed; but others, like the conversion vector table (CVT), need not be defined for CICS to function successfully. The resource table names are abbreviated to three-letter acronyms. Throughout the rest of this chapter, the acronym is used to refer to the tables, unless otherwise stated. Resources may reside on the local CICS/400 system, or on a remote CICS system. All resources residing on other CICS systems, and the systems themselves, should be defined in the appropriate resource definition table, before CICS can access the systems and use those resources. The tables are described briefly in Table 2 on page 43.

Resource definition architecture

Resource definitions and their groups relate to the OS/400 file structure. For more information about the OS/400 file structure, see the File Management topic in the iSeries Information Center.

The resource definitions are held as records within physical files. Each table has its own physical file and logical file. Installations should use their normal backup and recovery mechanism with these files. The file names and functions are as follows:

File name	Function
AAEGDACVP	Physical file containing part of CVT entry
AAEGDACFP	Physical file containing key information part of CVT entry
AAEGDACSP	Physical file containing selection criteria part of CVT entry
AAEGDADCP	Physical file containing DCT entries
AAEGDAFCP	Physical file containing FCT entries
AAEGDAGLP	Physical file containing GLT entries
AAEGDAJCP	Physical file containing JCT entries
AAEGDAPCP	Physical file containing PCT entries
AAEGDAPPP	Physical file containing PPT entries
AAEGDASIP	Physical file containing SIT entries
AAEGDASCP	Physical file containing TCS entries

AAEGDATCP	Physical file containing TCT entries
AAEGDATSP	Physical file containing TST entries
AAEGDACVL	Logical file relative to AAEGDACVP
AAEGDACFL	Logical file relative to AAEGDACFP
AAEGDACSL	Logical file relative to AAEGDACSP
AAEGDADCL	Logical file relative to AAEGDADCP
AAEGDAFCL	Logical file relative to AAEGDAFCP
AAEGDAGLL	Logical file relative to AAEGDAGLP
AAEGDAJCL	Logical file relative to AAEGDAJCP
AAEGDAPCL	Logical file relative to AAEGDAPCP
AAEGDAPPL	Logical file relative to AAEGDAPPP
AAEGDASIL	Logical file relative to AAEGDASIP
AAEGDASCL	Logical file relative to AAEGDASCP
AAEGDATCL	Logical file relative to AAEGDATCP
AAEGDATSL	Logical file relative to AAEGDATSP

Table 2. Resource definition table descriptions

Table	Description
CVT	<p>Conversion Vector Table</p> <p>The CVT controls any data conversion that may take place for the following functions:</p> <ul style="list-style-type: none"> • Interval control • File control • Temporary storage • Transient data • Link
DCT	<p>Destination Control Table</p> <p>The DCT defines intrapartition and extrapartition transient data storage queues. Four types of storage queue are supported. They are:</p> <ul style="list-style-type: none"> • Local intrapartition destinations that exist on the local CICS/400 system • Remote destinations residing on another system • Indirect destinations that point to another destination • Extrapartition destinations that exist outside the local CICS/400 system <p>Intrapartition queues support automatic task initiation (ATI); for more information, see <i>CICS for iSeries Application Programming Guide</i>.</p>
FCT	<p>File Control Table</p> <p>The FCT defines all the files accessed by EXEC CICS file control commands.</p>
GLT	<p>Group List Table</p> <p>The GLT identifies the library and resource definition groups that are to be installed when the control region is started.</p>
JCT	<p>Journal Control Table</p> <p>The JCT defines the CICS/400 user journals, and their access characteristics.</p>

Table 2. Resource definition table descriptions (continued)

Table	Description
PCT	<p>Program Control Table</p> <p>The PCT defines the identifiers of the transactions that may be run on the system. For each local transaction identifier, a program must also be specified in a PPT entry.</p>
PPT	<p>Processing Program Table</p> <p>The PPT defines the attributes of the user program objects that the CICS system can use.</p> <p>An entry must be made for each program and BMS map set that the system uses.</p>
SIT	<p>System Initialization Table</p> <p>The SIT defines the parameters that define the control region.</p>
TCS	<p>Terminal Control System Table</p> <p>The TCS contains the definitions of the remote systems with which CICS communicates.</p>
TCT	<p>Terminal Control Table</p> <p>The TCT defines all display devices and printers that are owned and used by transactions on the local CICS/400 system.</p> <p>Model terminal definitions can also be defined that are used by transactions routed from other systems. Model terminal definitions are used to generate TCT entries dynamically at runtime. This facility, known as autoinstall, is described in Chapter 11, "Autoinstall for terminal definitions" on page 305.</p>
TST	<p>Temporary Storage Table</p> <p>The TST is used to identify temporary storage queues. All remote temporary storage queues must be defined in the TST, as must those that need to be recoverable.</p>

To understand how the groups relate to the OS/400 file structure and how the definitions are stored, refer to Figure 6 on page 45, which shows that resource definition entries are records within members of OS/400 physical files. The names of the members of these physical files are the group names.

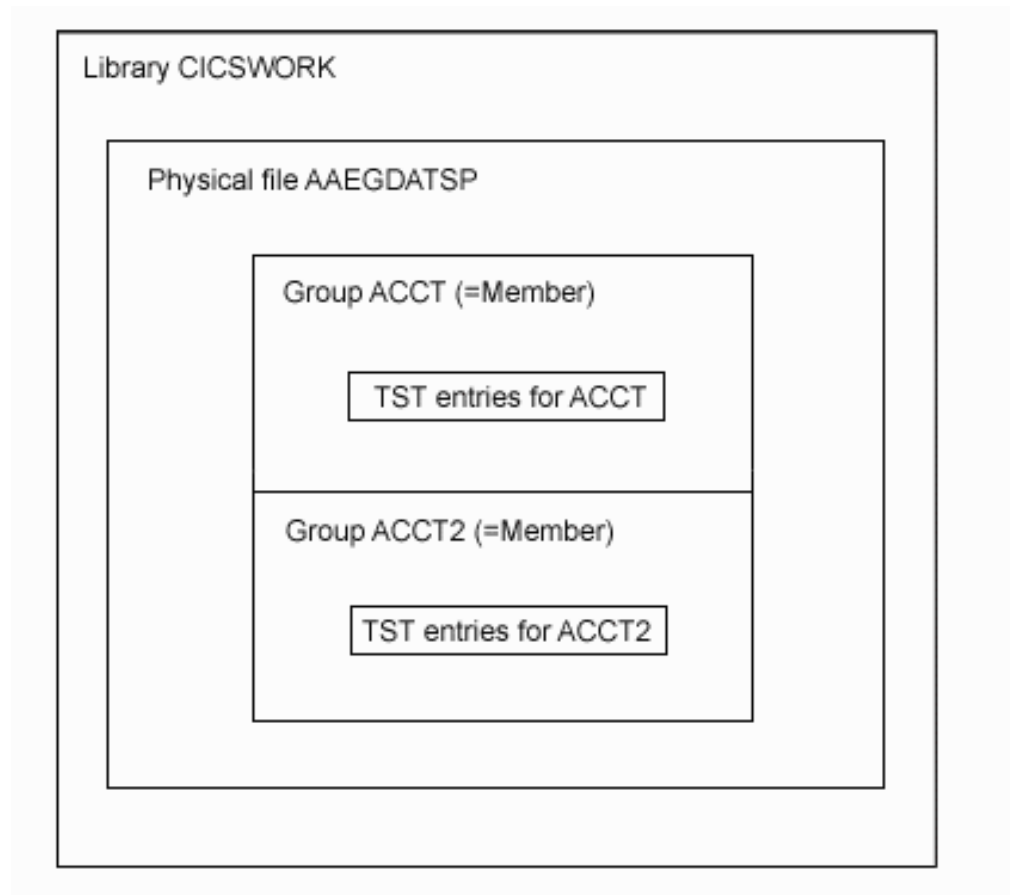


Figure 6. Relationship between resource definition and file structure

Creating resource definitions

Resource definitions for a control region should be created as follows:

1. Create a group using the CRTICSGRP command
2. Create the resource definitions for that group
3. Repeat steps 1 and 2 until all required groups and resources have been defined
4. Create a group list table (GLT) definition pointing to all groups required for the control region
5. Define a system initialization table (SIT) that points to the GLT

The GLT and SIT are special tables that are used once only in a control region. See “Specifying which groups are used” on page 47 for further information.

Defining and maintaining group definitions

You cannot create resource definitions until you have created the resource definition groups. When you create a group, a file is created for each type of table. You do not need to supply resource definitions for each table. In particular, the SIT and GLT might not contain entries.

What is a resource definition group?

A resource definition group defines a set of related resources. All table entries must be in a group. For example, resources may be grouped by application set or by business department. Other groups may include a set of common resources such as system definitions. This provides a way of selectively including resource definitions at control region startup. Predefined groups are provided for CICS-supplied transactions.

When a group is created, all resource definition tables, including a SIT and GLT, are created for that group. However, you do not have to provide entries for all tables. In particular, you may not want a SIT or GLT for all groups.

It is possible to assign a resource to more than one group. This means that different sets of resources can be specified for different situations. Thus resources with the same entry name but different group name may have differing data associated with them.

Using the resource definition CL commands

There are five CL commands used to create and maintain CICS groups:

CRTCICSGRP

Create a new group or recover a damaged group. A group must be created before you can add resource definitions to it.

When you create a group, one of two things happens:

1. If there are no other CICS groups in the specified target library, new physical files (for example AAEGDADCP) and logical files (for example, AAEGDADCL) will be created in the new library. All possible physical and logical files are created in the library even if you have no intention of adding resources of all types to the CICS group.
2. Alternatively, if a CICS group already exists in the target library, a new member is added to all the physical and logical files.

CHGCICSGRP

Use the CHGCICSGRP command to modify the descriptive text associated with a group. No other changes to the group definition are possible using this command. See "Using the CHGCICSGRP command" on page 137 for more details of this command.

DLTCICSGRP

Use the DLTCICSGRP command to remove all objects associated with a specific group. See "Using the DLTCICSGRP command" on page 138 for more details of this command.

SAVCICSGRP

Use the SAVCICSGRP command to save groups for use on either the current or previous release of CICS/400. SAVCICSGRP converts resource-definition tables for use on the previous release if necessary. This command calls the OS/400 SAVOBJ system command.

WRKCICSGRP

Use the WRKCICSGRP command to list a selection of groups and perform various operations on them. You can work with one or several groups at a time. The groups are selected according to criteria entered in the command. You can use either of the following methods:

- Enter the WRKCICSGRP command with selection criteria. You can change, delete, or work with any of the groups, or you can add new group definitions.
See “Using WRKCICSxxx commands” on page 54 for instructions on how to use the WRKCICSGRP command.
- Use the CEDA transaction.
See “Using CEDA resource definition online” on page 58 for instructions on how to use the CEDA transaction.

These commands are described in Chapter 10, “Defining resources-reference information” on page 133.

Specifying which groups are used

The SITLIB and SITGRP parameters of the STRCICS (start a CICS/400 control region) command used to specify the SIT to be used for the control region. See Chapter 8, “Administering the control region” on page 107 for details of the STRCICS command. The GLTLIB and GLTGRP parameters of the SIT specifies which GLT is to be used to load resource definitions. See “Using the ADDCICSSIT command” on page 239 for details of the ADDCICSSIT command. The INSLIB and INSGRP parameters of the Group List Table (GLT) entries define which resource groups are to be installed, and in which order, for that control region. The groups are installed in the order in which the GLT entries were created. If there are duplicate definitions, the entries in the later groups overwrite those already installed. If any of the groups named in the GLT include a GLT or SIT, these are ignored. Only the SIT named in the STRCICS command and the GLT named in that SIT are used in the control region.

The group containing the GLT may include other resource definition tables. If this is the case, the GLT group name must be included as an entry in the GLT. Otherwise the resource definitions will not be installed when the control region is started.

For example, the command:

```
STRCICS CTLRGN(TEST) SITLIB(JOHN) SITGRP(ALAN)
```

will start control region TEST using the SIT in group ALAN in library JOHN. The SIT contains the parameters GLTGRP(FRED) and GLTLIB(JOHN). CICS/400 uses the SIT in group ALAN to start up the control region and the GLT in group FRED to load resource definitions. The GLT contains the following group definitions:

```
ADDCICSGLT LIB(JOHN) GROUP(FRED) INSLIB(JOHN) INSGRP(ALAN)  
ADDCICSGLT LIB(JOHN) GROUP(FRED) INSLIB(JOHN) INSGRP(FRED)
```

When control region TEST starts, the GLT specified in the SIT, namely that in group FRED in library JOHN, is used to determine which resource definitions are installed. Groups ALAN and FRED are installed, with group ALAN being installed

first. Any definitions in group FRED that duplicate those in group ALAN overwrite those installed from group ALAN. If the GLT entry for group FRED had been omitted from the GLT, no resources in that group would have been installed, but the GLT in group FRED would still have been used to control which resources were installed.

Recovering groups

Use the **Recover** option of the **Work with CICS Group** panel. See “Using WRKCICSxxx commands” on page 54. The action taken depends upon two parameters of the CRTICISGRP command:

1. The RECOVER(*YES) parameter specifies that CICS/400 is to re-build the complete set of OS/400 members required for a specific group.

Note: This does not recover the records in the OS/400 members; it only makes sure that all the OS/400 members for the specified group exist.

2. The CLRGROUP parameter is valid only if RECOVER(*YES) is specified. This parameter checks that all OS/400 members for the specified group exist and removes any entries that you had made for that group. The group is left in the state that it was when first created; that is, with no entries.

Attention: Be careful how and when you use the CLRGROUP parameter. This parameter deletes all definitions in all table types for this group.

See “Using the CRTICISGRP command” on page 135 for more information about the RECOVER and CLRGROUP parameters.

Installing a group

Installing a group is achieved in one of the following ways:

- Using the ADDCICSGLT command
- Using the INSCICSGRP CL command
- Using the CEDA Install group function

Using the ADDCICSGLT command

Using the ADDCICSGLT command to include a group in a group list table and entering the group name in the INSGRP parameter. The group will then be installed, with the other groups in the GLT, at control region initialization.

The SITLIB and SITGRP parameters of the STRCICS command determine which SIT is used at control region initialization. The SIT parameters GLTLIB and GLTGRP (see “Using the ADDCICSSIT command” on page 239 for descriptions) name the GLT to be used. All the resource definition groups named in the GLT are installed for the control region.

Note: Remember that if you want to install the resources defined in the group specified in the GLTGRP parameter, you must include that group in its own GLT. See “Using the ADDCICSGLT command” on page 194 for more information.

Using the INSCICSGRP command

The INSCICSGRP CL command is used to install resources dynamically for an already-active control region. Unlike automatic resource installation at control region startup, dynamic resource installation is limited to the installation of a

single resource group. The INSCICSGRP CL command, which is run as an OS/400 batch job, installs a resource definition group into the control region named in the CTLRGN parameter.

You can install dynamically both new and amended resource definition entries for the following tables:

- Conversion vector table
- Program control table
- Processing program table
- Terminal control system table
- Terminal control table

You may install a maximum of 200 amended entries of each of these table types; there is no limit to the number of new table entries. All the entries must be in the same group.

The group is not installed if any resource in the group is being used; for example, if a terminal referred to by a TCT entry has an ACQUIRED status, or if a transaction referred to by a PCT entry is being used by someone on the system. You should take care that the terminal you are using is not included in the group you are installing. If it is, the installation will fail.

You can install new entries, but not amended entries, for the following tables:

- Destination control table
- File control table
- Journal control table

(To install amended entries for a file control table, you could use CEMT to disable and then discard the file entries.)

The INSCICSGRP CL command ignores the following tables in the group:

- Group list table
- System initialization table
- Temporary storage table

If the CONVERT parameter of the STRCICS command has been set to *PROMPT, you will be asked to specify whether or not you want any previous-release resource-definition files to be converted to the current release. See “Starting a control region (STRCICS)” on page 112 for details of the STRCICS CONVERT parameter.

Using the CEDA Install group function

The **Install group** function of the CEDA transaction is used to install resources dynamically for an already-active control region. Unlike automatic resource installation at control region startup, dynamic resource installation is limited to the installation of a single resource group.

The **Install group** function of the CEDA transaction may be used in one of the following ways:

- Fill in the options on the CEDA panel and press F13.
- Start a user shell using the STRCICSUSR command. When the screen clears, type:
CEDA INSTALL library group

where *library* is the name of the OS/400 library in which the group exists, and *group* is the name of the resource definition group to be installed.

- From the command line, enter a command of the form:
STRCICSUSR CTLRGN(name) TRANID(CEDA) DATA('INSTALL library group')

For details about the form of the STRCICSUSR command, see “Starting a user shell (STRCICSUSR)” on page 125.

You can install dynamically new and amended resource definition entries for the following tables:

- Conversion vector table
- Program control table
- Processing program table
- Terminal control system table
- Terminal control table

You may install a maximum of 200 amended entries for each of these table types; there is no limit to the number of new entries. All the entries must belong to the same group.

The group is not installed if any resource in the group is being used; for example, if the terminal referred to by a TCT entry has an ACQUIRED status, or if a transaction referred to by a PCT entry is being used by someone on the system. You should take care that the terminal you are using is not included in the group you are installing. If it is, the installation will fail.

You may install new entries, but not amended entries, for the following tables:

- Destination control table
- File control table
- Journal control table

(To install amended entries for a file control table, you could use CEMT to disable and then discard the file entries.)

The **Install group** function of CEDA ignores the following tables in the group:

- Group list table
- System initialization table
- Temporary storage table

If the CONVERT parameter of the STRCICS command has been set to *PROMPT, you will be asked to specify whether or not you want any previous-release resource-definition files to be converted to the current release. See “Starting a control region (STRCICS)” on page 112 for details of the STRCICS CONVERT parameter.

Successful completion of the CEDA **Install group** function results in the installation into the control region (being used by the user shell) of the group referred to in the CEDA transaction.

Note: CEDA is available only if you have defined the supplied transaction group AEGSPI. See “Defining supplied transactions” on page 65 for more information.

Defining and maintaining individual resource definitions

As with the CL commands that enable you to work with groups, there is a set of CL commands that enable you to define and maintain individual resource definitions. You must create the group before you create any resource definitions for the group.

The CL commands can be invoked by any of the following methods:

- From outside a user shell by:
 - Entering them on the OS/400 command line; see “Entering CL commands from the command line” on page 52 for instructions.
 - Coding them into an OS/400 CL program. An example is described in Chapter 3, “Installing CICS/400” on page 21.
- From inside a user shell using the supplied transaction CEDA. CEDA also gives you the ability to apply the changes to the control region while it is running. See “Using CEDA resource definition online” on page 58 for instructions on how to use the CEDA transaction.

Note: You cannot work with a table when CEDA is being run remotely using the CICS-supplied transaction CRTE, as it will be when using the CICS client terminal emulator. To process resource definitions, you need to interact with the OS/400 interface, which you cannot do through CRTE.

The commands are:

ADDCICSxxx

Add a resource definition

The OS/400 CL commands that enable you to add resource definition entries are called ADDCICSxxx.

Depending on the type of resource being entered, only certain parameters are presented. For example, if you are adding a definition for a local file with ADDCICSFCT, you are not prompted for the remote file name, record length, or key length.

Note: In a CL program environment, you are able to enter all the parameters that are available on a specific command; but when you run the program only the appropriate parameters take effect. Using the example above, if you had coded the remote file name, record length and key length, they would have been ignored if RMTSYSID(*NONE) was specified.

CHGCICSxxx

Change a resource definition

Existing definitions can be changed using the CHGCICSxxx commands. Each parameter has the reserved option of *SAME which means that if left unchanged the value that was held previously in the entry is used. As with the ADDCICSxxx functions, only parameters that are applicable to the type of entry that you are changing are displayed.

DSPCICSxxx

Display or print a resource definition

The DSPCICSxxx functions allow you to display all the values set for a selected group of entries. Alternatively, if you select OUTPUT(*PRINT), a spool file is created.

RMVCICSxxx

Remove an existing resource definition

WRKCICSxxx

Work with a resource definition table type

Use the WRKCICSxxx command to work with a number of definitions held in a specific table type. You can enter selection criteria by which entries are chosen for display. See "Using WRKCICSxxx commands" on page 54 for instructions on how to use the WRKCICSxxx command.

In each of these commands, *xxx* is the three-letter acronym for a resource table, for instance TCT for Terminal Control Table.

How to use the resource definition commands

You can use these commands in a number of ways. Outside CICS/400, you use the standard OS/400 command line and **Work with** panels. Within a CICS/400 shell, you use the CICS-supplied transaction CEDA.

Entering CL commands from the command line

The straightforward way to enter a command, if you are an experienced OS/400 user, is to use the command line on the OS/400 User Menu screen. You should include at least all the required parameters. Refer to the appropriate syntax diagrams in Chapter 10, "Defining resources-reference information" on page 133 for help, or press F1.

Using the prompt screens

If you are not so familiar with the command syntax, you can use prompt screens to help you complete commands. You start by typing the command on the command line, optionally followed by parameters.

Displaying a prompt screen

To display a prompt screen, press F4 Prompt at any point during entry of the command. If you type just the command and press F4, the screen shows only the required parameters for the command. Enter the required parameters and press Enter to display more fields.

Otherwise, the fields that are displayed when you press F4 depend on how many parameters you have already typed on the command line and what those parameters were. You see only the fields that are relevant to the type of resource being defined. Figure 7 on page 53 shows the screen you might see if you were defining a PCT entry for the supplied sample transaction ACCT. This screen lists acceptable values for each field and each field contains its default value. Note that you need a corresponding PPT entry for program ACCT00.

```

Add CICS PCT Entry (ADDCICSPCT)

Type choices, press Enter.

Library . . . . . CICSWORK_      Name, *LIB, *CURLIB
Group . . . . . ACCT_____      Name
Transaction . . . . . ACCT_____  Name
CICS system . . . . . *NONE_____ Name, *NONE
CICS program . . . . . ACCT00____  Name

Additional Parameters
Status . . . . . *ENABLED_____ *ENABLED, *DISABLED
Can be purged while executing . *YES      *YES, *NO
Maximum deadlock waittime . . . 0_____  0-7000
Maximum device I/O wait time . . 0_____  0-7000
Dump when Abend . . . . . *YES      *YES, *NO
Transaction work area size . . . 0_____  0-32767
Screen size used . . . . . *DFT      *DFT, *ALT
Local system queuing . . . . . *YES      *YES, *NO

More...
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

```

Figure 7. An example prompt screen

Displaying parameter keywords

If you press F11, the screen changes to show the parameter keywords, as they are in the syntax diagrams in Chapter 10, “Defining resources-reference information” on page 133, as well as the parameter description; see Figure 8. Press F11 again to change back.

```

Add CICS PCT Entry (ADDCICSPCT)

Type choices, press Enter.

Library . . . . . LIB              CICSWORK_
Group . . . . . GROUP            ACCT_____
Transaction . . . . . TRANSID     ACCT
CICS system . . . . . SYSID       *NONE
CICS program . . . . . PGMID      ACCT00____

Additional Parameters

Status . . . . . TRANSTS         *ENABLED_
Can be purged while executing . PURGE  *YES
Maximum deadlock waittime . . . WAITIME 0_____
Maximum device I/O wait time . . IDLETIME 0_____
Dump when Abend . . . . . DUMP    *YES
Transaction work area size . . . TWASIZE 0_____
Screen size used . . . . . SCRNSIZE *DFT
Local system queuing . . . . . LCLQUEUE *YES

More...
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

```

Figure 8. The example prompt screen showing parameter keywords

Entering resource and transaction identifiers

CICS/400 allows you to use lowercase letters and hexadecimal codes, in addition to uppercase letters, numbers and symbols, to specify **Resource identifier (RSRCID)** parameters in the CVT, **Transaction (TRANSID)** parameters in the DCT, PCT, and PPT, and **Remote transaction identifier (RMTTRANSID)** parameters in the PCT.

You should enclose any lowercase letters in apostrophes. Any lowercase letters not in apostrophes are converted to uppercase. Enter an ampersand (&) in the first character position to get an extended field size that allows for the entry of additional characters. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. For example, 'ab g' will be treated as 'ab' in the control region.

Hexadecimal characters should be enclosed in apostrophes and preceded by an uppercase X, for example X'A1A2A3A4'. Any hexadecimal value is accepted, but, if the value is not a printable character, the entry will be rejected when it is defined to a control region. Any EXEC CICS commands referencing the resource will return a NOTFND exception condition.

Getting further parameters

You can display more parameters in two ways:

- If a parameter can be repeated, for example the selection criteria parameters for a CVT entry, you will see **More...** at the bottom of the parameter list. Press F8 to display a screen showing the repeated fields.
- If there are more parameters for the command, F10 is listed with the function keys. Press it to display the next panel of parameters.

Getting help

To get help with a parameter, position the cursor on a field and press F1. A panel giving you help information about the field is displayed. This information is essentially the same as that in Chapter 10, "Defining resources-reference information" on page 133. Note, however, that you cannot display the command syntax diagrams.

Using WRKCICSxxx commands

The WRKCICSxxx commands allow you to perform operations on a selection of resources. If you are an experienced OS/400 user, you will find the WRKCICSxxx process and panels familiar.

Working with CICS groups

To access the **Work with CICS group** panel, type in the WRGCICSGRP command followed by a library name. Optionally, you can include a group name.

For example, suppose you typed in the following command:

```
WRKCICSGRP LIB(CICSWORK) GROUP(ACCT)
```

The screen displayed would look similar to Figure 9 on page 55.

The release level of each resource definition file is checked. An error message is issued if OS/400 and CICS/400 are at different release levels or if the resource definition file is at an unsupported CICS/400 release level. If the resource

definition file is at the previous CICS/400 release level, the message **Previous release group** is displayed in the **Description** field.

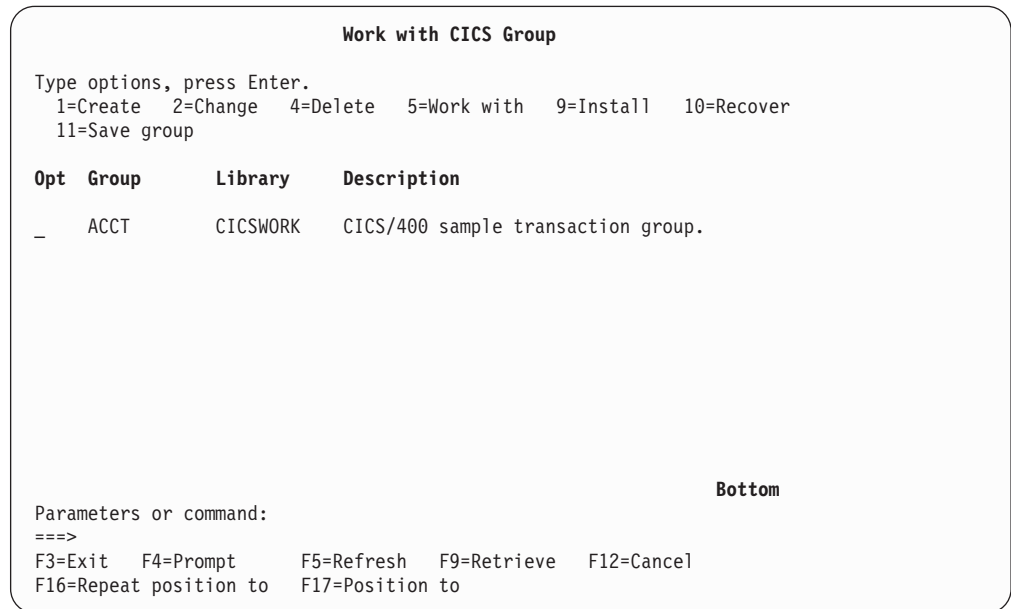


Figure 9. The Work with CICS group screen

You can perform any of the options listed on the screen shown in Figure 9, by typing the associated number on the row showing the required group. Table 3 describes each option.

Table 3. Work with CICS group screen options

Option	Description
1	Create a new group. This is equivalent to entering a CRTICISGRP command (see "Using the CRTICISGRP command" on page 135).
2	Change the text describing a group. This is equivalent to entering a CHGCICISGRP command (see "Using the CHGCICISGRP command" on page 137).
4	Delete a group. This is equivalent to entering a DLTCICISGRP command (see "Using the DLTCICISGRP command" on page 138). You are asked to confirm the deletion before it takes place.
5	Work with the individual resources for that table within the selected group. This is equivalent to entering a WRKCICISxxx command.
9	Install a group. This is equivalent to entering an INSCICISGRP command (see "Using the INSCICISGRP command" on page 138).
10	Recover a damaged group. See "Recovering groups" on page 48 for information about recovering a group.
11	Save a group. This option is part of the release-to-release compatibility function. See "Release-to-release compatibility" on page 17 for information about release-to-release compatibility and "Using the SAVCICISGRP command" on page 140 for information about saving a group.

If you select option 5, to work with individual resource definition types within a specific group, you are presented with another panel,

Working with tables

You access the **Work with CICS Tables** panel by selecting Option 5 on the **Work with CICS group** panel. See Figure 10, which lists the 11 different table types.

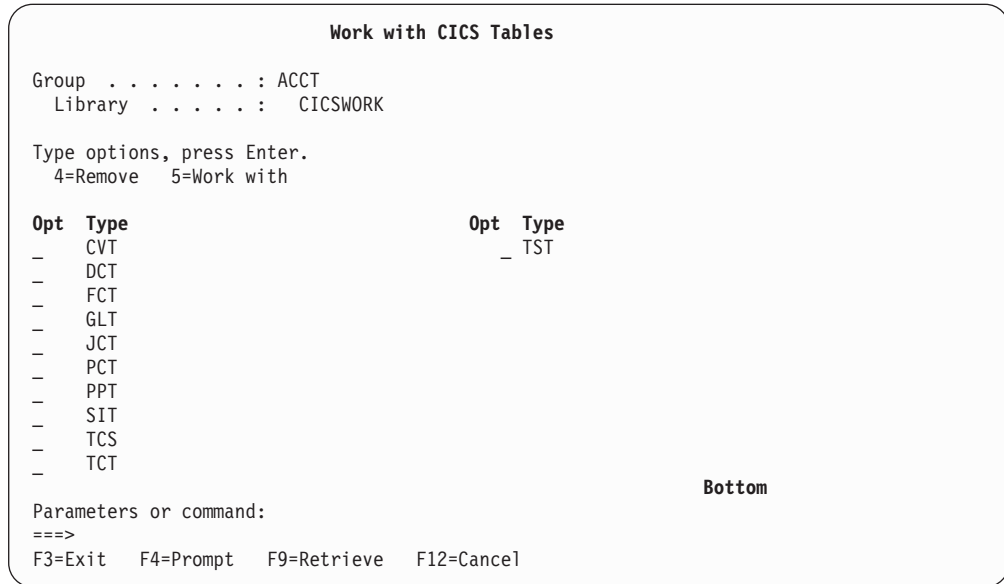


Figure 10. The Work with CICS tables screen

Enter one of the option numbers beside one of the tables. The options are described in Table 4.

Table 4. Work with CICS tables screen options

Option	Description
4	Remove from the table resource definitions belonging to the selected group. This is equivalent to entering a RMVCICStxxx command to delete all resource definitions.
5	Work with the individual resources for that table within the selected group. This is equivalent to entering a WRKCICStxxx command.

Working with resource definitions

You can select definitions to work with in two ways:

- By selecting Option 5 from the **Work with CICS Tables** panel.
- By typing in a WRKCICStxxx command on the command line.
For example, to work with all PCT definitions in group ACCT, type the following:

```
WRKCICSPCT LIB(CICSWORK) GROUP(ACCT)
```

If no matching entry to the selection criteria is found, then an error message is displayed.

Figure 11 on page 57 shows an example of a screen that you might see if you chose to work with PCT definitions for the supplied sample application ACCT.

```

Work with CICS PCT

Library . . . . . : CICSWORK   Group . . . . . : ACCT

Type options, press Enter.
 1=Add  2=Change  3=Copy  4=Remove  5=Display  6=Print

Opt  Transaction      CICS      CICS      Remote
   Transaction      program   system    transaction  Status
-    -              -         -         -
-    ACCT            ACCT00    *NONE     *TRANSID    *ENABLED
-    ACEL            ACCT03    *NONE     *TRANSID    *ENABLED
-    ACLG            ACCT03    *NONE     *TRANSID    *ENABLED
-    AC01            ACCT01    *NONE     *TRANSID    *ENABLED
-    AC02            ACCT02    *NONE     *TRANSID    *ENABLED
-    AC03            ACCT03    *NONE     *TRANSID    *ENABLED
-    AC05            ACCT03    *NONE     *TRANSID    *ENABLED
-    AC06            ACCT03    *NONE     *TRANSID    *ENABLED
-                                     Bottom

Parameters or command:
===>
F3=Exit  F4=Prompt  F5=Refresh  F9=Retrieve  F12=Cancel
F16=Repeat position to  F17=Position to

```

Figure 11. An example Work with CICS PCT screen

From the **Work with CICS xxx** screen, you can choose one of the options by putting the appropriate number in the **Opt** field of one or more selected entries. The options are described in Table 5.

Table 5. Work with CICS xxx screen options

Option	Description
1	This option can be placed only in the extended action list row of the Work with panel. This is the first (blank) row displayed at the top of the list and is always present even if no entries were found to match the required selection criteria. This option allows you to add a new entry to the list. Note: To get the new entry displayed in the list you have to “Refresh” the panel by using the F5 function key.
2	This option allows you to change an existing entry. Note: If you use the CHGCICSxxx function from the OS/400 CL command line, all the parameters are presented with the value *SAME. If you use the CHGCICSxxx function from the WRKCICSxxx panel, the current value for the parameter is substituted before you begin to make your changes.
3	This option allows you to copy an existing entry. With the Change option, the previous values for the parameters are substituted before you make any changes. Note: As with the Add option, you must refresh the panel before the new entry is displayed.
4	This option allows you to remove an obsolete entry.
5	This option allows you to display all the values set for a specific entry.
6	This option allows you to print all the values set for a specific entry.

Note: Modifications to all resource definitions do not take effect in an active control region until either the control region is restarted or the resource definitions have been installed dynamically. See “Installing a group” on page 48 for more information.

Using CEDA resource definition online

The CEDA transaction is used to maintain the resource definitions, and to apply any changes, while CICS/400 is running.

The transaction is started by entering CEDA on the command line and pressing Enter.

You will see a panel like that in Figure 12.

The CEDA panel is a front end for the CL commands. Select a table by typing a

```

Resource Definition Online

Library . . . . .: _____
Group . . . . .: _____

Type option, press Enter.
  1=Select
  Table      Description
  _CVT      Data Conversion Table
  _DCT      Destination Control Table
  _FCT      File Control Table
  _GLT      Group List Table
  _JCT      Journal Control Table
  _PCT      Program Control Table
  _PPT      Processing Program Table
  _SIT      System Initialization Table
  _TCS      Terminal Control System Table
  _TCT      Terminal Control Table
  _TST      Temporary Storage Table

Enter  F1=Help  F3=Exit  F12=Cancel  F13=Install group

```

Figure 12. CEDA transaction screen

library name, a group name, and a 1 by the required table. Press the Enter key. If you are authorized to work with that table, the WRKCICSxxx command is invoked and you will see a panel like that in Figure 11 on page 57. See “Working with resource definitions” on page 56 for information on how to continue.

When you have finished changing entries and returned to the CEDA transaction screen, Figure 12, you should install the resources using the F13 key. The changed resources will be installed dynamically and made available in the current CICS/400 system.

The function keys available to you from the CICS transaction screen shown in Figure 12 are described in Table 6.

Table 6. CEDA function keys

Key	Description
F1	The HELP key. It also lists all the function keys and their function.
F3	Ends the CEDA transaction.
F12	Resets the CEDA panel.
F13	Invokes the Install group function. The group must be specified. The group is checked to see if it exists, you must have authority to use it, and there must be at least one table associated with it. This is equivalent to entering an INSCICSGRP command. (See “Using the INSCICSGRP command” on page 138.)

Note: If you use CRTE to run CEDA remotely, as for example when using the CICS client terminal emulator, you can use only the Install group function; you cannot work with a table. To process resource definitions, you need to interact with the OS/400 command interface, which you cannot do through CRTE.

Chapter 5. Defining a basic control region

The IVP described in Chapter 3, “Installing CICS/400” on page 21 runs with the supplied resource definitions and system initialization table (SIT). This chapter explains the steps that were performed to define these resources for the IVP control region. The procedures are the same for any control region. By stepping through the setup of the control region for the sample application, you can gain an understanding of the setup process and how to start defining your own control region.

There is a description of the control region in Chapter 8, “Administering the control region” on page 107.

The topics covered in this chapter are:

- “Defining the system initialization table (SIT)”
- “Defining temporary storage and transient data files” on page 63
- “Defining supplied transactions” on page 65
- “Setting default wait times” on page 67
- “Setting job priorities” on page 67
- “Storage considerations” on page 67

The control region is identified by a four-character control region identifier (CTRLGN). This four-character identifier is used with the STRCICSUSR command to associate a user shell with a control region. It is also used by other CICS systems to communicate with this CICS system. This ID is defined at control region startup in the CTRLGN parameter of the STRCICS command. It is also used in the temporary storage and transient data file names.

Security

To use the functions and commands required to set up a control region, you require one of the following:

- Security officer authority
- Access to the appropriate commands, granted using the GRTOBJAUT command

For further information about security, see Chapter 6, “Security requirements for CICS/400” on page 75, and the *iSeries Security Reference*.

Defining the system initialization table (SIT)

The system initialization table (SIT) defines the overall characteristics of the entire system, rather than specific application or resource concerns. This section steps through the creation of the SIT for the IVP.

The SIT definition is added to the group that was created using the CRTICSGRP command (see page 29). The ADDCICSSIT command required for the IVP is as follows:

```
ADDCICSSIT LIB(CICSWORK) GROUP(ACCT) GLTLIB(CICSWORK)
           GLTGRP(ACCT) WRKARASIZE(100) DUMP(*NO)
           DEVCTL(3457) INTTRCCTL(7000 *YES *NO)
```

```

AUXTRCCTL(*YES *NO CICSWORK/SAMPTRACE1
CICSWORK/SAMPTRACE2) USRTRC(*YES) TSCTL(0
*NO) SHRSTG((400 20 10) (400 20 10) (400
20 10)) NONSHRSTG((200 20 10) (200 20 10)
(200 20 10)) TDCTL(*INVOKER *NO)

```

Table 7 describes the various parameters in the sample ADDCICSSIT command. See “Using the ADDCICSSIT command” on page 239 for reference information about this command.

Table 7. Parameters on the sample ADDCICSSIT command

Parameter	Explanation
ADDCICSSIT LIB(CICSWORK) GROUP(ACCT)	Add a SIT entry to group ACCT in library CICSWORK. (A member called ACCT now exists in file CICSWORK/AAEGDASIP.)
GLTLIB(CICSWORK) GLTGRP(ACCT)	When this SIT is referred to in a STRCICS command, the GLT member ACCT contains the list of groups of resource definitions to be installed at startup. See page 48 for more information about how correct GLT is acquired at control region initialization.
WRKARASIZE(100)	The control region being started with this SIT will have a common work area size of 100 bytes.
DUMP(*NO)	System dumps are not allowed.
DEVCTL(3457)	Autoinstalled terminal identifiers consist of the fourth, fifth, sixth, and eighth characters of the device ID. The digits are identified using an offset relative to zero.
INTRCCTL(7000 *YES *NO)	The internal trace can hold 7000 entries, it is active at control region startup, and it does not wrap when full.
AUXTRCCTL(*YES *NO CICSWORK/SAMPTRACE1 CICSWORK/SAMPTRACE2)	The auxiliary trace facility is active at startup, the trace file does not switch when full, and the user space objects that the auxiliary trace will use are named.
USRTRC(*YES)	User trace entries are created at control region startup.
TSCTL(0 *NO)	During a warm or emergency start, temporary storage queues are deleted. (But, if in-doubt units of work are affecting the queues, see “Setting the STRTYPE parameter” on page 114 for the results.)
SHRSTG((400 20 10) (400 20 10) (400 20 10))	Shared storage requirements. This is allocated for the control region using this SIT. The shared space object will be created to hold 400KB. Ten extents of 20KB each are allowed for each of these storage types. Refer to “Storage considerations” on page 67 for more information about shared storage requirements.
NONSHRSTG((200 20 10) (200 20 10) (200 20 10))	Nonshared storage requirements. This is allocated for each shell activated under the control region using this SIT. The nonshared storage objects for each shell will be created to hold 200KB. Ten extents of 20KB each are allowed for each storage type. Refer to “Shell storage objects” on page 72 for more information about nonshared storage requirements.
TDCTL(*INVOKER *NO)	The user profile for a CICS transaction started as a result of a transient data trigger is the USERID that caused the trigger level to be reached. During a warm or emergency start, transient data queues are deleted. (But, if in-doubt units of work are affecting the queues, see “Setting the STRTYPE parameter” on page 114 for the results.)

Defining temporary storage and transient data files

You must define two physical files to hold auxiliary temporary storage and intrapartition transient data queues. These files are required, even if you do not define any DCT or TST entries for your CICS system.

- One file for recoverable temporary storage and transient data queues
- One file for nonrecoverable temporary storage and transient data queues

CICS/400 requires that these files be created with specific file names in the library that will be used when the control region is started. The required file names are:

File	Description
AAEGxxxTN	Non-recoverable file
AAEGxxxTR	Recoverable file

The *xxxx* in the file names should be replaced by the value in the CTLRGN parameter of the STRCICS command used to start the control region. For the IVP, the file names should be AAEGSAMPTN and AAEGSAMPTR for the example control region.

See Chapter 7, “Defining your own control region” on page 81 for more information about temporary storage and transient data.

File characteristics

Each file must be a keyed file with the following characteristics:

- A key length of 16 bytes
- A variable-length record size
- A maximum record size of 32 742 bytes

You should remember that the maximum length of temporary storage and transient data records is restricted to 20 bytes less than the maximum record size for these files. These 20 bytes are used to hold header information for the record.

The maximum temporary storage length, in turn, may affect any CICS function that uses temporary storage internally. For example, interval control uses temporary storage to hold any data supplied with an EXEC CICS START command. The length of this data is restricted to 98 bytes less than the maximum temporary storage record length, the extra 98 bytes being used to store interval-control header information.

In addition, the recoverable file AAEGSAMPTR must be defined to OS/400 commitment control. For details, see “Recoverable file commitment control requirements” on page 104.

Creating the physical files

File objects QCICS/AAEGCICSTR (recoverable) and QCICS/AAEGCICSTN (nonrecoverable) are installed with CICS/400. You can copy these objects by using the CRTDUPOBJ command to create the file in the required library and with the proper authority level. If you copy the file, the single member name in each file has to be changed to be the same as the file name. Alternatively, you can use Data Description Specifications (DDS) to create the files as desired. See the DDS Reference PDF in the iSeries Information Center for OS/400 utility information.

Examples of the CRTDUPOBJ command to copy the recoverable and nonrecoverable files from the QCICS library are shown in Figure 13. These example commands copy the supplied temporary storage and transient data files for control region SAMP. In addition, when the files are copied, you should change the file attributes to reflect the storage requirements for control region SAMP.

```

CRTDUPOBJ OBJ(AAEGCICSTR) FROMLIB(QCICS) OBJTYPE(*FILE) TOLIB(MYLIB)
NEWOBJ(AAEGSAMPTR)

CHGPF FILE(MYLIB/AAEGSAMPTR) SIZE(100000 1000 8)
TEXT('CICS/400 TS/TD RECOVERABLE file for CR SAMP')

CRTDUPOBJ OBJ(AAEGCICSTN) FROMLIB(QCICS) OBJTYPE(*FILE) TOLIB(MYLIB)
NEWOBJ(AAEGSAMPTN)

CHGPF FILE(MYLIB/AAEGSAMPTN) SIZE(100000 1000 8)
TEXT('CICS/400 TS/TD NONRECOVERABLE file for CR SAMP')

```

Figure 13. Using the CRTDUPOBJ command

The DDS equivalent to using the CRTDUPOBJ command for these two files are shown in Figures 14 and 15.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	00010A* TS/TD RECOVERABLE FILE (AAEGctlrgnTR)	
200	00020A	
300	00030A	
400	00040A R AAEGSAMPTR	UNIQUE
500	00050A RECLEN 2B 0	TEXT('TS/TD Recoverable record')
600	00070A QUEID 8A	TEXT('Record Length')
700	00060A QUETYP 2B 0	TEXT('Queue Id')
800	00080A ITEMNUM 2B 0	TEXT('Type of Queue')
900	00090A QUEOWN 1A	TEXT('Item Number')
1000	00100A RESERVE1 3A	TEXT('Queue Owner')
1100	00110A RECIMAGE 32722A	TEXT('Reserved')
1200	00120A	TEXT('Record Image')
1300	00130A K QUEID	VARLEN
1400	00120A K QUETYP	
1500	00140A K ITEMNUM	
1600	00150A K QUEOWN	
1700	00160A K RESERVE1	

Figure 14. DDS for TS/TD recoverable file AAEGSAMPTR

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	00010A*	TS/TD NON-RECOVERABLE FILE (AAEGc1trgnTN)
200	00020A	
300	00030A	
400	00040A	R AAEGSAMPTN UNIQUE TEXT('TS/TD Non-Recoverable record')
500	00050A	RECLEN 2B 0 TEXT('Record Length')
600	00070A	QUEID 8A TEXT('Queue Id')
700	00060A	QUETYP 2B 0 TEXT('Type of Queue')
800	00080A	ITEMNUM 2B 0 TEXT('Item Number')
900	00090A	QUEOWN 1A TEXT('Queue Owner')
1000	00100A	RESERVE1 3A TEXT('Reserved')
1100	00110A	RECIMAGE 32722A TEXT('Record Image')
1200	00120A	VARLEN
1300	00130A	K QUEID
1400	00120A	K QUETYP
1500	00140A	K ITEMNUM
1600	00150A	K QUEOWN
1700	00160A	K RESERVE1

Figure 15. DDS for nonrecoverable file AAEGSAMPTN

Defining supplied transactions

The CICS-supplied transactions are defined in special groups, as shown in Tables 8 and 9. In these tables, **Program** indicates that it is specified in the supplied CICS transaction definition, and **Group** indicates the supplied CICS resource definition group that contains the CICS transaction definition.

The transactions given in Table 8 have an operator interface.

Table 8. Supplied transactions with an operator interface

Transaction ID	Program	Group	Description
CEBR	AEGBRTSS	AEGEDF	Browse CICS temporary storage queues. See the <i>CICS for iSeries Application Programming Guide</i> .
CECI/CECS	AEGCIPGM	AEGINTE	CICS command-level interpreter/syntax checker. See the <i>CICS for iSeries Application Programming Guide</i> .
CEDA	AEGDAPGM	AEGSPI	CICS resource definition online. See page 58.
CEDF	AEGDFFLG	AEGEDF	CICS execution diagnostic facility. See the <i>CICS for iSeries Application Programming Guide</i> .
CEMT	AEGCMDRV	AEGOPER	CICS master terminal facility. See page 325.
CESF	(internal)	(internal)	Signoff. See page 129.
CRTE	AEGTRRTE	AEGISC	CICS transaction routing. See page 351.

The supplied transactions given in Table 9 do not have an operator interface. If you enter these transaction identifiers, you will cause an abend and maybe system failure. (Use of the synchronization-level and client-related transactions is described in *CICS for iSeries Intercommunication*.)

Table 9. Supplied transaction groups with no operator interface

Transaction ID	Program	Group	Description
CATD	AEGTCATD	(internal)	Autoinstall delete
CMPX	AEGFSMXP	AEGISC	CICS local queuing shipped
CRSR	AEGTRROU	AEGISC	CICS relay transaction
CVMI	AEGFSMIR	AEGISC	Synchronization-level 1 mirror
CPMI	AEGFSMIR	AEGISC	Synchronization-level 1 mirror
CSMI	AEGFSMIR	AEGISC	Synchronization-level 2 mirror
X'02' (CSM2)	AEGFSMIR	AEGISC	Synchronization-level 2 mirror
X'03' (CSM3)	AEGFSMIR	AEGISC	Synchronization-level 2 mirror
CCIN	AEGAPCIN	AEGCLI	CICS Client INstall transaction
CTIN	AEGAPCIN	AEGCLI	CICS Terminal INstall transaction
CSRR	AEGTRFIB	AEGISC	Transaction routing relay program
CSSF	(internal)	(internal)	Signoff for transaction routing

To incorporate supplied transactions into a control region, use the INSGRP parameter to add the supplied transaction group to the group list table (GLT) to be used when you start your control region. You do not have to create program control table (PCT) entries for these transactions. The INSLIB parameter must be QCICS for supplied-transaction groups. The group list table is specified in the **GLT group** parameter of the SIT; see “Defining the system initialization table (SIT)” on page 61 and “Using the ADDCICSSIT command” on page 239 for further details. See “Using the ADDCICSGLT command” on page 194 for a description of the ADDCICSGLT command.

This example assumes that the ACCT control region requires all the supplied transactions. Figure 16 shows an ADDCICSGLT command adding each supplied transaction group to a GLT called ACCT in library CICSWORK.

```

ADDCICSGLT LIB(CICSWORK) GROUP(ACCT)
INSLIB(QCICS) INSGRP(AEGCLI)

ADDCICSGLT LIB(CICSWORK) GROUP(ACCT)
INSLIB(QCICS) INSGRP(AEGEDF)

ADDCICSGLT LIB(CICSWORK) GROUP(ACCT)
INSLIB(QCICS) INSGRP(AEGINTER)

ADDCICSGLT LIB(CICSWORK) GROUP(ACCT)
INSLIB(QCICS) INSGRP(AEGISC)

ADDCICSGLT LIB(CICSWORK) GROUP(ACCT)
INSLIB(QCICS) INSGRP(AEGOPER)

ADDCICSGLT LIB(CICSWORK) GROUP(ACCT)
INSLIB(QCICS) INSGRP(AEGSPI)

```

Figure 16. Example GLT entries for CICS-supplied transactions

Setting default wait times

CICS/400 uses OS/400 data queuing for the information exchange between shells and a control region. Because this involves data queue waits, CICS/400 jobs are affected by the **default wait time** setting for the job class in which the job is executed. It is important that the default wait time *not* be set to zero. A preferred setting is 30 seconds for interactive CICS jobs (like user shells) and 120 seconds for batch jobs (like the control region).

For further information, see *Work Management*.

Setting job priorities

By definition, a CICS system is a transaction processing system executing short-term, mostly interactive, applications. Because of this, the control region job should run at an equal or higher priority than its shell processes. Your installation setup and application requirements will help you to determine the proper job priorities for your individual CICS systems.

For further information, see *Work Management*.

Storage considerations

This section discusses issues that need to be considered in order to tune storage requirements to your control region and application or user shells.

CICS/400 provides support to manage storage requests for both internal CICS and user application requirements.

CICS/400 uses three types of user space objects to manage storage requests:

1. Local space object (AEGxxxxLSO)
Located in OS/400 system domain storage and used to maintain and keep track of CICS storage elements in both system and user storage objects.
2. System storage space object (AEGxxxxSYS)
Located in OS/400 system domain storage and used specifically for internal CICS storage requirements.
3. User storage space object (AEGxxxxUSR)
Located in OS/400 user domain storage, and obtained and managed by CICS. CICS uses this space when servicing storage requests on behalf of applications.

Note: Throughout this section, the *xxxx* in object names should be replaced by the value of the CTLRGN parameter on the STRCICS command; this is SAMP for the IVP.

CICS/400 makes a distinction between shared and nonshared storage:

- Shared storage is accessible across many CICS address spaces (control region and its associated shells). Shared storage contains common data that is needed by CICS service and user application modules. Shared storage areas (or space objects) are acquired by the control region at CICS startup. They are released when the control region shuts down.
- Nonshared storage is only accessible to an individual CICS address space and contains data that is specific to that address space. Nonshared storage areas (or

space objects), used for managing storage elements within the shell, are acquired by each CICS shell (user or batch) when it is started. They are released when the shell shuts down.

Control region storage objects

At control region initialization, the control region manager creates the three types of user space object in the QTEMP library for that control region. For each storage space object, the SHRSTG parameter in the SIT is made up of three elements. Each element defines:

- The initial size of the space object
- The size of each extent of the object, when the initial size is reached
- The maximum number of extents to allow for the object

Within the SHRSTG parameter:

- Element 1 defines the local space object (LSO)
- Element 2 defines the system space object (SYS)
- Element 3 defines the user space object (USR)

For information about the ADDCICSSIT or CHGCICSSIT CL commands, refer to “Using the ADDCICSSIT command” on page 239 and “Using the CHGCICSSIT command” on page 248.

For an example of a SIT showing sample settings for the SHRSTG parameter, see “Defining the system initialization table (SIT)” on page 61.

The values specified in the SHRSTG parameter are used at control region startup to create the user space objects. The extent-related parameters are used, when necessary, during the lifetime of the control region, to extend the space to fulfill storage requests from within CICS.

Estimating control region storage requirements

The defaults set within the SHRSTG parameter in the SIT are set large enough to allow the initialization of only a limited-function CICS control region. For example, using the default setting, you would *not* be able to initialize a control region with the CECI supplied transaction, because the default size for the AEGSAMP SYS object is not large enough for CECI’s internal requirements.

To calculate the amount of storage needed for the control region space objects, you need a lot of information about the control region being set up and the application requirements of that control region. If the applications in the control region perform many shared storage requests (for example, shared GETMAIN requests or many WRITE requests to MAIN temporary storage), you will need more storage. The AEGSAMPUSR space object should be created to handle the peak loads, without causing many CICS tasks to wait to obtain shared storage.

The amount of free space available within the space objects fluctuates during the lifetime of the control region. The type of processing being done by the CICS applications and the current mix of applications affects the free space availability. After some experience with a particular control region and the applications within it, you should be able to adjust the storage requirements to suit the control region.

Note that CICS will automatically extend these space objects as required during processing. Therefore, it is not essential that these parameters be specified exactly. The balance should be to allow CICS to make initial allocations that are large enough for the usual configurations so that CICS is not required to spend extra

time providing new allocations while still allowing these spaces to grow as required. Initially, if you set a lower initial value and allow for several extensions, you can then let CICS allocate the storage that is needed to run your applications. You can then look at the size of each of these space objects in the QTEMP library of the control region job and see how much space is actually required. You can then adjust the initial allocations and increments to get better performance.

The control region AEGSAMPLSO space object

The SHRSTG defaults for the AEGSAMPLSO object (in the ADDCICSSIT command) are set large enough to handle the requirements for a minimum-sized control region. The AEGSAMPLSO object contains the CICS internal storage management information. CICS/400 manages its control region storage requests using this space object.

The size requirements for the AEGSAMPLSO depend heavily on:

- The amount and the type of storage used in the USR and SYS control region space objects.
- The resource table requirements, and on the requirements of the supplied transactions.

Guidance on estimating these storage requirements is given in “The control region AEGSAMPSSYS space object”.

The following guideline can be used to calculate an approximate size for the AEGSAMPLSO space object. The amount calculated will probably need adjustment as the needs of the control region being defined become clearer over time.

AEGSAMPLSO size

```
Estimated AEGSAMPLSO size = 4096 +
(100 * (total number of resource table entries defined))
+
(100 * (total number of CICS internal control blocks))
```

The control region AEGSAMPSSYS space object

The AEGSAMPSSYS space object of the control region contains all the shared resource table entries. In addition, it contains many other CICS internal control blocks, such as those used by specific CICS components, and those used by certain supplied transactions.

Resource table sizes: Note that there are some basic storage sizes and calculations you can use to guide the creation of the storage objects. Each CICS runtime resource table entry is of a fixed size. The resource table entries are all allocated within the AEGSAMPSSYS object of the control region. The sizes given in Table 10 represent the approximate size of each table entry. CICS stores each table entry on a 16-byte boundary. The alignment may cause more storage to be used than is indicated by the individual size.

Table 10. Size required for resource table definitions

Resource table	Size
CVT	Portion of record: <ul style="list-style-type: none"> • Key area - 16 bytes • Data area - 286 bytes
DCT	144 bytes

Table 10. Size required for resource table definitions (continued)

Resource table	Size
FCT	108 bytes
JCT	76 bytes
PCT	68 bytes
PPT	112 bytes
TCT	272 bytes
TCS	80 bytes
TST	32 bytes

Internal storage requirements: The control region creates many internal control blocks in the system shared object (AEGSAMPSYS). These are created and deleted as required by CICS. Those of most interest to a system administrator are included in Table 11 with an explanation of their use.

Table 11. Internal control blocks created

Storage usage	Size (bytes)	Description
Extrapartition TD queues	48	Created for each TD file opened by the control region.
Interval control	98	Created for each interval-control start request.
Task control	30	Created for each unique ENQ request.
Dispatch control area	96	One exists for each active shell for a control region. There is always at least one of these per control region.
Automatic transaction initiation	22	Created for each scheduled ATI request.
Deferred work element	240	Created for deferred work to be processed by the control region.
Terminal control entries	208	Created for each active terminal. This is in addition to the TCT resource entry. In addition, one is created for each autoinstalled terminal.
Terminal system entries	368	Created for each active session to a remote CICS system. This is in addition to the TCS resource entry.
File anchor block	320	One exists for each file opened by CICS.
Temporary storage – main	varying	Each temporary storage main queue created as a result of an application.

CICS-supplied transactions: The CICS-supplied transactions also have some storage requirements. If you decide to include the supplied transactions in your control region, you need to be aware of these requirements. Table 12 illustrates the resource table requirements for the CICS-supplied transactions.

Table 12. CICS-supplied transaction groups storage requirements

Group	Number of PCT definitions (68 bytes each)	Number of PPT definitions (112 bytes each)
AEGCLI	2 x 68 = 136	1 x 112 = 112

Table 12. CICS-supplied transaction groups storage requirements (continued)

Group	Number of PCT definitions (68 bytes each)	Number of PPT definitions (112 bytes each)
AEGEDF	2 x 68 = 136	8 x 112 = 896
AEGINTER	2 x 68 = 136	10 x 112 = 1120
AEGISC	6 x 68 = 408	2 x 112 = 224
AEGOPER	1 x 68 = 68	44 x 112 = 4928
AEGSPI	1 x 68 = 68	1 x 112 = 112

In addition, certain of the supplied transactions require internal storage in the AEGSAMP SYS (control region system space object). Only those supplied transactions with specific storage requirements are listed in Table 13.

Table 13. Supplied transaction requirements

Supplied transaction	Group	Minimum size (bytes)
CECI/CECS	AEGINTER	151 000
CEDF	AEGEDF	1920

CICS common work area (CWA): If the control region is to have a common work area (CWA – indicated by the WRKARASIZE in the SIT), CICS creates the CWA in the AEGSAMP SYS object of the control region. The size of the CWA needs to be included in the size estimates for this space object.

Calculating the size of the system space object: The calculation that follows can be used to estimate the minimum size requirement for the AEGSAMP SYS space object. This calculation is provided only as a guideline and may not meet the requirements for your specific control region. Where internal control blocks are created or deleted during the lifetime of the control region, you should allow enough space to handle the peak loads for the system being defined.

AEGSAMP SYS size

```

Minimum AEGSAMP SYS size =
size of each CICS resource table entry defined
+
size of CICS component internal control blocks
+
size of common work area (CWA)
+
storage required for supplied transactions
  (defined to this control region)
+
size of temporary storage main queues
    
```

The control region AEGSAMPUSR space object

The SHRSTG defaults for the AEGSAMPUSR object (in the ADDCICSSIT CL command) are set large enough to handle the requirements for a minimum-sized control region. This space object is used for:

- GETMAIN requests issued internally within the CICS control region
- Shared GETMAIN requests on behalf of CICS applications
- Interval control requests for expired starts
- Intersystem communications requests

- Main temporary storage queues

If the control region uses many of the functions described in the list above, you should adjust the default settings to allow for more space to handle the peak loads for your control region.

Shell storage objects

At shell initialization, CICS creates the three types of user space objects in the OS/400 QTEMP library. For each storage space object, the NONSHRSTG parameter in the SIT is made up of three elements. Each element defines:

- The initial size of the space object
- The size of each extent of the object, when the initial size is reached
- The maximum number of extents to allow for the object

Within the NONSHRSTG parameter:

- Element 1 defines the local space object (LSO)
- Element 2 defines the system space object (SYS)
- Element 3 defines the user space object (USR)

For information about the ADDCICSSIT or CHGCICSSIT CL commands, refer to “Using the ADDCICSSIT command” on page 239 and “Using the CHGCICSSIT command” on page 248. For an example of a SIT showing sample settings for the NONSHRSTG parameter, see “Defining the system initialization table (SIT)” on page 61.

The values specified in the NONSHRSTG parameter are used at shell startup to create the user space objects. The extent-related parameters are used, when necessary, during the lifetime of the control region, to extend the space to fulfill storage requests from within CICS.

As with the control region, the amount of free space available within the shell space objects fluctuates during the lifetime of the shell. The type of processing being done by the CICS applications in the shell affects the free space availability.

The storage requirements for CICS user shells should be considerably less than the control region. The main factors used to determine the size of a shell space object are:

- The applications running within the shell
- The number of open files allowed
- The nonshared storage requirements of the applications

In addition, if you are using an autoinstall control program, your CICS/400 shells require 5KB more space to allow for autoinstall processing requirements. See Table 14 on page 73. Change the element 3 of the NONSHRSTG parameter of the SIT to allow for this extra storage.

Note that CICS will automatically extend these space objects as required during processing. Therefore, it is not essential that these parameters be specified exactly. The balance should be to allow CICS to make initial allocations that are large enough for the usual configurations so that CICS is not required to spend extra time providing new allocations while still allowing these spaces to grow as required. Initially, if you set a lower initial value and allow for several extensions, you can then let CICS allocate the storage that is needed to run your applications. You can then look at the size of each of these space objects in the QTEMP library

of the application job and see how much space is actually required. You can then adjust the initial allocations and increments to get better performance.

The shell AEGSAMPLSO space object

The NONSHRSTG defaults for the AEGSAMPLSO object (in the SIT) are set large enough to handle the requirements for a minimum-sized CICS shell.

The size requirements for the AEGSAMPLSO depends heavily on the amount of storage and the type of storage used in the other two shell space objects. CICS manages its shell nonshared storage requests using this space object. The AEGSAMPLSO object contains the CICS internal storage management information.

The following guideline can be used to calculate an approximate size for the AEGSAMPLSO space object. The amount calculated will likely need adjustment as the needs of the shell become clearer.

AEGSAMPLSO size

Estimated shell AEGSAMPLSO size = 4096 +
(100 * (total number of CICS shell control blocks))

Table 14 gives guidance on estimating the shell control block sizes.

The shell AEGSAMPSYS space object

The calculation that follows can be used to estimate the minimum size requirement for the AEGSAMPSYS space object. This calculation is provided only as a guideline.

AEGSAMPSYS size

Minimum AEGSAMPSYS size =
size of CICS component internal control blocks (used by the control region)
+
storage required for CEMT (if used)

Storage requirements for internal control blocks and for CEMT are described in the next sections.

Internal control block storage requirements: CICS creates many internal control blocks in the shell AEGSAMPSYS. These are created and deleted as required by the shell. Those of interest to a system administrator are included in Table 14 with an explanation of their use.

Table 14. Internal control blocks for a CICS/400 shell

Shell storage usage	Size (bytes)	Description
Task control	304	Created for each unique ENQ request for a specific resource within one CICS task.
Automatic transaction initiation	22	Created for each scheduled ATI request for a shell.
File anchor block	320	One exists for each file opened by the shell.
Automatic file close	32	One exists for each file opened by the shell and can be closed by the CICS automatic file close facility.

Table 14. Internal control blocks for a CICS/400 shell (continued)

Shell storage usage	Size (bytes)	Description
Terminal system entries	368	Created for each active session to a remote CICS system. Entry needed only for a shell doing ISC processing.
Autoinstall control program COMMAREA parameters	5000	Created to hold the parameter areas to be passed to the autoinstall control program AEGTCACP during terminal installation. This control block is created only if the control region has a PPT entry defined and enabled for AEGTCACP.

CICS-supplied transactions shell storage requirements: Many of the CICS-supplied transactions make use of shell storage for nonshared storage obtained by GETMAIN. This is allocated within the AEGSAMPUSR storage object for the shell.

The CEMT transaction creates and uses storage within the AEGSAMPUSYS object of the shell. This storage is used only when the CEMT transaction is being run within a shell, and is deleted by CICS when CEMT ends. If a shell is allowed to use the CEMT transaction, allowances need to be made for this storage in the AEGSAMPUSYS object of the shell. If you need to run the CEMT transaction within a shell, the minimum storage requirement is 7KB.

The shell AEGSAMPUSR space object

The NONSHRSTG default values for the AEGSAMPUSR object (in the ADDCICSSIT CL command) are set large enough to handle the requirements for a minimum-sized CICS shell. This space object is used mainly for nonshared GETMAIN requests issued within the shell, and for storing the COMMAREA from transaction to transaction. If you know that the applications use large COMMAREAs or large nonshared GETMAIN requests, you should adjust the default settings to account for this.

Summary

You have now reviewed the process of gathering information, translating that information into resource definitions, and starting a control region with that information. First, you need to review your security requirements; Chapter 6, "Security requirements for CICS/400" on page 75 gives you guidance. Then you can use the information in Chapter 7, "Defining your own control region" on page 81 to adapt the example IVP to create your own control region.

Chapter 6. Security requirements for CICS/400

CICS/400, in the same way as any other system running on an iSeries, needs the protection of a security mechanism to ensure that the resources it owns are protected from unauthorized access.

Security for CICS/400 is provided on the iSeries by the OS/400-supplied security facilities. There are no CICS-specific security facilities on the iSeries.

Resource security is the principal mechanism by which security can be applied to the CICS/400 system and the applications that run on it and interface with it. Many of the CICS/400 resource definitions map directly to the OS/400 objects. By applying OS/400 object authority restrictions, access to these resources, CICS/400 functions, and applications can be controlled. Although CICS/400 does not have its own security mechanism, the application programming interface receives the usual CICS response of NOTAUTH to unauthorized access as expected on other CICS platforms.

Command-level security, which restricts the use of the system programming commands such as EXEC CICS INQUIRE, EXEC CICS SET, EXEC CICS PERFORM, and EXEC CICS DISCARD, is not provided by CICS/400. The use of these commands may be controlled by resource-level security on the program objects that perform the commands. (See “Supplied transactions” on page 77.)

The following iSeries facilities can be used in setting up security control for CICS/400:

- Signon security and display station security
- Initial menu and initial program security
- OS/400 command security
- Resource security
- Adoptive program security
- Location security
- User profile and group profile security

Signon security and display station security

These mechanisms control which users may sign on to the iSeries and the terminals at which they may sign on. As users do not sign on within CICS, the user ID from the general iSeries logon is used to determine subsequent resource access authority. See *iSeries Security Reference* for further details.

The OS/400 has five levels of system security, that determine the level of resource security applied to users:

- | | |
|-----------------|--|
| Level 10 | The system does not require a password to sign on. Users have access to all system resources. |
| Level 20 | The system requires a password to sign on. Users have access to all system resources. |
| Level 30 | The system requires a password to sign on. Users must have authority to access objects and system resources. |

- Level 40** The system requires a password to sign on. Users must have authority to access objects and system resources. Programs fail with an object domain error if they try to access objects through interfaces that are not supported.
- Level 50** The system requires a password to sign on. Users must have authority to access objects and system resources. Programs fail if they try to pass unsupported parameter values to supported interfaces, or if they try to access objects through interfaces that are not supported.

Initial menu and initial program security

This mechanism is used to control which program is run when a user signs on, and which menu is presented to the user, after the initial program has run. Menu security can be used to prevent use of any OS/400 commands, including CICS-related ones, on the command line, either by not displaying a command line or by restricting commands through the user profile.

OS/400 command security

This comprises:

- Limited capability checking
- Resource checking of command program objects

These mechanisms can be used to prevent a user from controlling the running of a control region by restricting the use of the STRCICS and ENDCICS commands. The user can also be prevented from controlling the running of user shells by restricting the use of the STRCICSUSR and ENDCICSUSR commands. Related commands such as CRTICSCBL and PRTCICSTRC should also be considered in this category.

Limited capability checking

A user's profile may be set to indicate whether or not the user has limited capability. Any OS/400 command can have an attribute indicating whether or not it can be invoked by a limited-capability user. By this means, limited-capability users can be prevented from using certain commands.

If it is considered necessary to restrict further the use of certain commands, resource checking must be used. (See "Resource checking of command program objects".)

Resource checking of command program objects

Object authority may be used to control the use of CICS-related OS/400 commands where the limited capability facility is insufficient. For example, suppose that the STRCICSUSR and CRTICSCBL commands use limited capability to restrict general use. One user, however, needs the authority to run the STRCICSUSR command whilst being prevented from using the CRTICSCBL command. In this instance, the user may be given object authority to the STRCICSUSR command object (STRCICSUSR object type *CMD in library QCICS) but not to the CRTICSCBL command program object. Indeed, any user who does not need to issue any commands could be prevented from accessing all members of the QCICS library, but it must be considered that if a command is executed from within a program, the user must have authority to execute that command.

Resource security

This facility is the most important mechanism by which access to CICS/400 resources (programs, files, transient data queues, and so on) is controlled. (See “Resource checking of command program objects” on page 76 for a description of resource security in its use to restrict command usage.)

Resource security can be used to prevent particular types of access to a resource, for example, read, update, add, or delete. This may be applied to supplied transaction security.

Supplied transactions

For transactions that are defined in the control region, you should set the CICS/400 program objects to PUBLIC *EXCLUDE. Then each user needing access could be put in an authorization list for the object or could be given individual authority to use the object. The program objects for the supplied transactions are listed in “Defining supplied transactions” on page 65.

You can also prevent access to supplied transactions for a particular CICS system by not installing those transactions in the CICS control region. Do this by excluding the definitions for these transactions from the GLT used at control region startup.

User transactions

Access to user transactions can be controlled using the **Transaction (TRANSID)** parameter of the ADDCICSTCT command. (See “Using the ADDCICSTCT command” on page 275.) You can use this parameter to restrict the use of the terminal to a single named transaction.

Application example

As with the supplied transactions, there are various ways you can define security for a particular application. However, typically, resource security would be used as in the following example.

Examine the resource definition examples in Table 15 and the user profiles in Table 16.

Table 15. Resource definition example

PCT	PPT	Program object	File object used
TRN1	PROGRAM1	MYLIB/PROGRAM1	TRN1/MASTER TRN1/UPDATES

Table 16. User profiles and privileges

User profile	What the user can do
USER1	Can update all the files referred to by TRN1. Can use MYLIB/PROGRAM1.
USER2	Can only use the inquiry functions of TRN1. Can use MYLIB/PROGRAM1. Cannot update either of the files referred to by TRN1 program.
USER3	No access to any TRN1 functions. Cannot access the files by using the TRN1 transaction. No access to the files at all.

Using the examples, the following restriction could be set up for each user, using the GRTOBJAUT command:

- USER1 should be given *USE authority for the program, and *ALL authority for the files.
- USER2 should be given *USE authority for the program object, and the files.
- USER3 should be given *EXCLUDE authority and prevented from using program object MYLIB/PROGRAM1 and the two file objects.

The same approach applies to applications accessing the other object types, namely temporary storage queues, journals, programs, BMS maps, or intrapartition transient data queues. However, you should note that, for extrapartition transient data queues, the authority of the control region is used to determine access restrictions, not the authority of the user of the application in the user shell.

Adoptive authority

Adoptive authority indicates whether or not a program should run under the authority of the user (*USER) or the authority of the owner (*OWNER). See the *iSeries Security Reference* manual for further information.

The CRTICSCBL and CRTICSC CL commands do not support adoptive authority directly. All programs created with these commands have the *USER authority by default. To run CICS/400 application programs that include, for example, high-security system programming commands, and for which you require high authority, you can create a CL program (using a high-authority user profile) to call the CICS/400 application program, using the CRTCLPGM CL command, with the **User profile (USRPRF)** parameter set to *OWNER. The CL program runs under the higher-level *OWNER authority, as does the CICS/400 application program that it calls. Because the CL program has not ended, the CICS/400 program is running, in effect, under adoptive security.

Communication security

The aspects of security for systems communicating with each other using Advanced Program-to-Program Communication (APPC) are discussed in *APPC Programming*. See also the *CICS for iSeries Intercommunication* manual.

User profile and group profile security

An OS/400 user profile must be set up for every iSeries user. This allows the user access to the iSeries (running at security level 20 or above). It also holds authority information and default classes, that is *PGMR, *USER, and *SECADM. These classes give special authority to certain areas of security, namely, spool control, job control, and access to all objects existing on the OS/400. When a user creates a program or file object, the user has owner authority to that object, that is, full object rights. Only profiles with all object authority can have full authority to these objects. All other profiles have to be authorized using either the GRTOBJAUT command or the EDTOBJAUT command.

A group profile can be set up that has special authority or specific authority to certain objects. A user profile can then be set up that references a group profile.

The group profile is created using the CRTUSRPRF CL command, and an option on this command allows a user profile to refer to a group profile. For example, several employees working in the payroll department may need access to certain

critical files. You could give access to each object in each user profile. Alternatively, you could create a group profile that gives access to these objects and change the user profiles to refer to the group profile.

Chapter 7. Defining your own control region

Chapter 5, “Defining a basic control region” on page 61 describes how to set up a control region to run the IVP. Whether you have run the IVP, or defined a control region using the sample program DEFCICSRGN, you now need to adapt that information to apply to the control region that will run your transactions. This chapter describes, and provides examples for, the CICS information that you may need to:

- Define resources required to run a CICS system
- Create the OS/400 objects referred to by the resource definitions

In addition, this chapter tells you how to customize the definition information. In this way, you can adapt the system to meet the business requirements of your applications.

This chapter considers the main components that make up the system and file management functions of CICS/400, and what you need to set up a control region that is capable of handling the needs of your application users.

The following areas are considered:

- How do OS/400 file management considerations relate to CICS/400 resources? See “OS/400 data management considerations” on page 82.
- What programs, transactions and maps are to be used in your system? See “Transaction, program, and map considerations” on page 82.
- How can temporary storage queues be used with a CICS system? See “Temporary storage considerations” on page 84.
- How can transient data queues be used with a CICS system? See “Transient data considerations” on page 85.
- What type of file access and file organization will you be using with CICS? See “File control considerations” on page 88.
- Will this CICS system make extensive use of interval control processes? See “Interval control considerations” on page 94.
- Have there been any resources defined as recoverable in the CICS/400 system? See “Recovery considerations” on page 104.
- Does this CICS system have any CICS/400 journal control requirements? See “Journal control considerations” on page 96.
- Does this CICS system have many resources defined as remote to the local OS/400 system? See “Device considerations” on page 100.
- Are there any applications in this CICS system using printer spooling commands? See “Printer spooling considerations” on page 102.
- How should CICS trace be set up? See “Trace considerations” on page 104.

Security

To use the functions or commands required to set up a control region, you need one of the following:

- Security officer authority
- Access to the necessary commands, granted using the GRTOBJAUT command

For further information about OS/400 security, see Chapter 6, "Security requirements for CICS/400" on page 75, and the *iSeries Security Reference* manual.

OS/400 data management considerations

CICS/400 uses the native OS/400 file and commitment control facilities to ensure the integrity of data files. Native OS/400 file objects are opened by CICS user shells. CICS shells can share these resources and use native system services to maintain data integrity. CICS uses the following files types:

- User database files accessed by CICS application programs. Any number of user files, emulating VSAM access methods for key-sequenced data sets (KSDS), entry-sequenced data sets (ESDS), and relative-record data sets (RRDS), can be attached.
- A file for each control region that contains all recoverable transient data intrapartition and auxiliary temporary storage data queues.
- A file for each control region that contains all nonrecoverable transient data intrapartition and auxiliary temporary storage data queues.
- CICS user journal files (one OS/400 file for each journal).
- User printer spool files.
- Display files (for terminal input).
- Intersystem communications files (ICF).
- Extrapartition transient data queue files.

All the above CICS resources must be defined to CICS in the appropriate CICS resource definition table. See Chapter 4, "Defining resources" on page 41 for an introduction to resource definition, and Chapter 10, "Defining resources-reference information" on page 133 for reference information. Additionally, the system administrator has to create some of the corresponding file objects before their use within a CICS application.

Notes:

1. CICS/400 does not support DDM files.
2. OS/400 file expiration dates have no effect when the files are used under CICS/400.

Transaction, program, and map considerations

A CICS application consists of one or more programs written to perform a specific function. A particular invocation of such a program is known as a transaction and is identified by a transaction identifier (TRANSID). You tell CICS how you want the transaction to run in a PCT definition. In the PCT definition you specify the program to be invoked by the transaction and options related to functions provided by CICS.

The program associated with a transaction is defined in the PPT. The PPT entry contains control information for the program, such as the program object that will be utilized by the program identifier and the programming language used.

Each interactive application using a display device can use screen layouts or maps. These are created using basic mapping support (BMS). (See the *CICS for iSeries Application Programming Guide* for information about BMS.) An application may use a series of related maps at different times during the interaction with the user. These related maps must belong to a map set. Even if your program has only one map, it must still belong to a map set. Each map set must have a PPT definition with the CICS MAP parameter specified as *YES. There is no link between a program and its map sets. The map set name must be coded in the BMS EXEC CICS SEND MAP and EXEC CICS RECEIVE MAP commands in your program. (See the *CICS for iSeries Application Programming Guide* for information about these commands.)

There are no OS/400 requirements for defining CICS transactions to a CICS/400 system. However, you should ensure that the program objects and map user spaces used by transactions are defined to CICS in a library that is accessible to the users of these objects. Also, you should follow the security guidelines in Chapter 6, "Security requirements for CICS/400" on page 75.

Example PCT definitions

Figure 17 shows an example PCT entry for a transaction called TRN. The transaction is disabled. The entry refers to a PPT entry called TRNPRGRM.

```
ADDICSPCT LIB(CICSWORK) GROUP(GROUP2) TRANSID(TRN) TRANSTS(*DISABLED)
          PURGE(*NO) PG MID(TRNPRGRM) SCRNSZE(*ALT) LCLQUEUE(*NO)
```

Figure 17. Example PCT entry for a disabled transaction

Figure 18 shows an example PCT entry for a transaction called TRN1. The transaction is enabled and refers to a PPT entry called TRN1PGM.

```
ADDICSPCT LIB(CICSWORK) GROUP(ACCT) TRANSID(TRN1)
          PG MID(TRN1PGM) DUMP(*NO)
```

Figure 18. Example PCT entry for an enabled transaction

Example PPT definitions

Figure 19 shows an example of a PPT entry for a map called MAP1M. Because this entry defines a map, the CICS MAP parameter must be *YES. Also, MAP1M must be a user space object within the control region library list.

```
ADDICSPPT LIB(CICSWORK) GROUP(ACCT) PG MID(MAP1M) CICS MAP(*YES)
          PG M OBJ(MAP1M)
```

Figure 19. Example PPT entry for a BMS map

Figure 20 on page 84 shows a PPT entry for a program PROG1. The CICS MAP parameter may be omitted as this entry is defining a program and CICS MAP(*NO) is the default. PROG1 must be a program object within the control region library list.

```
ADDICSPPT LIB(CICSWORK) GROUP(ACCT) PGMID(PROG1) PGMOBJ(PROG1)
```

Figure 20. Example PPT entry for a program object

Temporary storage considerations

Temporary storage (TS) provides a scratchpad area for holding data created by one transaction, to be used later by the same transaction or by a different transaction.

Temporary storage queues remain intact until they are deleted by either the originating task or another task. They can be accessed any number of times in any order. Even after a task has terminated, any temporary storage it originated and left undeleted, is still available to other tasks.

Temporary data can be stored either in main storage or in auxiliary storage:

- Main storage

Generally, main storage should be used if the data is needed for short periods of time and does not require recovery. Data in main storage does not survive from one CICS run to the next. Main storage might be used to pass data from task to task.

- Auxiliary storage

In CICS, auxiliary storage is a physical file, and auxiliary storage should be used to store large amounts of data or data needed for a long period of time. Data stored in auxiliary storage is retained after CICS termination and can be recovered in a subsequent restart. In CICS, data in auxiliary storage can be designated as either recoverable or nonrecoverable. A recoverable temporary storage queue needs an entry in the TST for backout purposes. However, even nonrecoverable queues may be restored during either a warm or an emergency restart, depending on the setting of the TSCTL parameter in the SIT.

In addition, temporary storage queues can be defined in the TST as remote to the CICS system. This means that the actual data for the remote TS queues exists in another CICS system.

For more information about application programming aspects of temporary storage, see the *CICS for iSeries Application Programming Guide*.

Recoverable and nonrecoverable TS queues

If you have defined CICS/400 temporary storage queues as **recoverable**, the information about these queues is kept in the TS/TD recoverable file AAEGxxxxTR for backout purposes. The TS/TD recoverable file is also used by CICS/400 to recover these queues during a warm or emergency restart.

You must also define a physical file AAEGxxxxTN to hold nonrecoverable queues. Further information about these files is in “Defining temporary storage and transient data files” on page 63.

Example TST definitions

An example TST entry for a recoverable temporary storage queue is shown in Figure 21. The definition applies to all temporary storage queues in group ACCT with names beginning with AU.

```
ADDCICSTST LIB(CICSWORK) GROUP(ACCT) TSQUEUE(AU) TYPE(*RECOVERABLE)
```

Figure 21. Example TST entry for a recoverable temporary storage queue

An example of a TST entry for the recoverable temporary storage queue TSRECOVR is shown in Figure 22.

```
ADDCICSTST LIB(CICSWORK) GROUP(ACCT) TSQUEUE(TSRECOVR) TYPE(*RECOVERABLE)
```

Figure 22. Example TST entry for a recoverable temporary storage queue

An example TST entry for a remote temporary storage queue is shown in Figure 23.

```
ADDCICSTST LIB(CICSWORK) GROUP(GROUP2) TSQUEUE(Queue2) TYPE(*REMOTE)  
SYSID(SYS1)
```

Figure 23. Example TST entry for a remote temporary storage queue

Transient data considerations

Transient data (TD) provides a generalized queuing facility. Data can be queued (stored) for subsequent internal or external processing. You can route data either to or from a predefined symbolic destination; the destinations may be either intrapartition or extrapartition.

Destinations are intrapartition if associated with a facility allocated to the CICS control region, and extrapartition if the data is directed to a destination that is external to the CICS control region. The destinations must be defined in the destination control table (DCT) when the CICS control region is initiated.

You can define two types of destinations:

- **Intrapartition destinations** are queues of data for use with one or more programs running as separate tasks. You can direct data to or from these destinations.
- **Extrapartition destinations** are queues (files) residing on any sequential device. In general, sequential extrapartition destinations are used for storing and retrieving data outside the CICS control region. For example, one task may read data from a remote terminal, edit the data, and write the results to a physical file for subsequent processing in another control region.

In addition, intrapartition and extrapartition destinations can be defined as remote to the CICS system. This means that the actual data for these destinations exists in another CICS system.

Intrapartition and extrapartition destinations can also be used as indirect destinations. Using indirect destinations:

- Provides the programmer with some flexibility because data can be routed to one of several destinations by an entry in the DCT.
- Minimizes the need for changes to application programs when a destination changes.

For further information about transient data, see the *CICS for iSeries Application Programming Guide*.

Extrapartition transient data

Extrapartition transient data queue files are shared by all user shells in the control region. These files are opened at control region startup (or by the EXEC CICS SET TDQUEUE command in application programs), not opened by the individual user shells. All accesses to these files are processed by CICS transient data service modules running in the control region.

When starting the control region, you must have authorized access to the OS/400 objects referred to by extrapartition TD requests. In addition, all the OS/400 objects must have been created before control region startup, using the appropriate OS/400 command for creating the object. The STGDEV parameter for the DCT definition defines the type of storage on which the TD queue resides; the appropriate OS/400 command should be used to define the object:

CRTPF	DASD
CRTPRTF	Printer
CRTTAPF	Tape
CRTDKTF	Diskette

See the ADDCICSDCT command (“Using the ADDCICSDCT command” on page 161) for more information.

For extrapartition queues, CICS/400 uses the maximum record length defined for the underlying OS/400 object as the record length.

Extrapartition physical files can be registered to an OS/400 journal, but the control region cannot open them under OS/400 commitment control. Therefore, an extrapartition transient data queue cannot be recoverable.

Recoverable intrapartition TD files

If you have defined intrapartition transient data queues as recoverable, the information in these queues is kept in the TS/TD recoverable file AAEGxxxxTR for backout purposes. The TS/TD recoverable file is also used by CICS/400 to recover these queues during a warm or emergency restart.

You must also define a physical file AAEGxxxxTN to hold nonrecoverable queues. Further information about these files is in “Defining temporary storage and transient data files” on page 63.

Note: You will not be able to start the control region until the TS and TD physical file requirements are complete.

Example DCT definitions

An example DCT definition for an extrapartition transient data queue is shown in Figure 24 on page 87. The queue name is BETA. The queue maps to OS/400

physical file CICSWORK/AACXTRA(AAXCTRA), which should have been created as a file with variable-length or fixed-length records.

```
ADDICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(BETA) TYPE(*EXTERNAL)
          FILE(AACXTRA) MBR(AAXCTRA)
```

Figure 24. Example DCT entry for an extrapartition transient data queue

An example DCT entry for an intrapartition transient data queue is shown in Figure 25. The queue name is SAMA. The trigger level is 55, and transaction TRAN will be started automatically when the number of records in the queue reaches that figure.

```
ADDICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(SAMA) TYPE(*INTERNAL) TRGLVL(55)
          TRANSID(TRAN)
```

Figure 25. Example DCT entry for an intrapartition transient data queue

An example DCT entry for a remote transient data queue is shown in Figure 26. The queue name is RMT1. The remote queue is held on system SYS1 and has a record length of 512 bytes.

```
ADDICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(RMT1) TYPE(*REMOTE) SYSID(SYS1)
          LENGTH(512)
```

Figure 26. Example DCT entry for a remote transient data queue

An example DCT entry for an indirect destination is shown in Figure 27. Destination IND1 in group ACCT in OS/400 library CICSWORK, is defined with an indirect queue type. The physical destination identifier is RMT1, which has already been defined in the DCT as a remote destination.

```
ADDICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(IND1) TYPE(*INDIRECT)
          PHYDEST(RMT1)
```

Figure 27. Example DCT entry for an indirect destination

Defining a CSMT log destination

The CSMT log is an optional transient data destination to which control region and shell status messages, including transaction abend messages, are written. This facility does not affect the usual handling of transaction abend messages, which are also written to the user shell and displayed on the terminal running the transaction.

You define the CSMT log by adding a DCT entry with a DEST parameter value of CSMT to a group that is installed in your control region. The CSMT destination may be intrapartition, extrapartition, or indirect; it cannot be remote.

An example DCT definition for an intrapartition CSMT transient data destination is shown in Figure 28 on page 88. An intrapartition definition for CSMT must specify a non-recoverable transient data queue.

```
ADDCICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(CSMT) TYPE(*INTERNAL)
```

Figure 28. Example DCT entry for an intrapartition CSMT transient data queue

An example DCT definition for an extrapartition CSMT transient data destination is shown in Figure 29. This example uses a member of a disk file to hold the CSMT output. The CSMT destination could also be a printer file.

```
ADDCICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(CSMT) TYPE(*EXTERNAL)  
FILE(CICSWORK/TDQUEUES) MBR(CSMT) OPENOPTION(*OUTPUT)  
RCDFMT(*VARIABLE)
```

Figure 29. Example DCT entry for an extrapartition CSMT transient data queue

For this example you need to create an OS/400 physical file (CICSWORK/TDQUEUES) with a variable length record format and a suggested minimum record length of 160 bytes. Do this using the CL command CRTPF as indicated in Figure 30.

```
CRTPF FILE(CICSWORK/TDQUEUES) SRCFILE(CICSWORK/QDDSSRC)  
SRCMBR(TDQUEUES) MBR(CSMT)
```

Figure 30. Example of creating a physical file to hold CSMT output

An example DDS for the physical file for this extrapartition CSMT destination is shown in Figure 31.

```
00010A* TDQUEUES - EXTRAPARTITION TDQ - VARIABLE LENGTH  
00020A  
00030A          R TDQUEUES          TEXT('TDQUEUES')  
00040A          RECIMAGE          160A          TEXT('Record Image')  
00050A          VARLEN
```

Figure 31. Sample DDS in CICSWORK/QDDSSRC.TDQUEUES

See “Using the ADDCICSDCT command” on page 161 for further details on defining transient data destinations. See *CICS for iSeries Problem Determination* for descriptions of possible abend codes.

File control considerations

CICS/400 uses the OS/400 Data Management facilities to control access to physical files for CICS/400 applications. CICS/400 service modules provide for the VSAM emulation by:

- Applying the VSAM rules to the file control EXEC CICS commands
- Using the FCT definition and the OS/400 file attributes to validate the CICS/400 access to the file
- Using the appropriate OS/400 Data Management access method for performing I/O for CICS/400 applications

Supported file types

Table 17 on page 89 provides a cross-reference between the VSAM access methods and the OS/400 equivalent.

Table 17. OS/400 equivalents of VSAM file access methods

VSAM	OS/400 equivalents
KSDS	A physical or logical file created through the use of DDS to define the file and the file keys. These types of files are accessed by key in CICS/400 applications.
ESDS	An arrival-sequence physical file created either through the use of DDS or by a create physical file (CRTPF) command. ESDS files can have either fixed-length or variable-length records. Variable-length record ESDS files can only be defined using DDS.
RRDS	An arrival-sequence physical file created either through the use of DDS or by a create physical file (CRTPF) command. Unlike ESDS files, RRDS files must have fixed-length records. RRDS files are preformatted. This is emulated on OS/400 by initializing the file with deleted records. Access to an RRDS file is by relative-record number (RRN).

CICS/400 supports alternate index access to KSDS and ESDS files. The OS/400 equivalent of an alternate index is a logical file. A logical file can only be created by DDS, and requires that the physical file and the DDS references must be created first.

If you are unfamiliar with these VSAM file types or are unclear about the use of keys for VSAM files, refer to the *CICS for iSeries Application Programming Guide*.

VSAM files accessed by CICS programs are attached and opened when first referred to by an EXEC CICS command or by a CICS service module. They remain attached to the shell until closed by one of the following methods:

- The automatic file closure facility. See “Automatic file closure”.
- The SET FILE CLOSED command of the CEMT supplied transaction. See “CEMT INQUIRE | SET FILE” on page 336.
- The EXEC CICS SET FILE CLOSED system programming command. See the *CICS for iSeries Application Programming Guide* for details.

Automatic file closure

CICS opens files on first reference within a user shell. Files can be closed in CICS by using either the CEMT SET FILE CLOSED supplied transaction or the EXEC CICS SET FILE CLOSED command.

In addition, CICS provides an automatic file closure facility. This facility provides for the closing of files within each user shell based on SIT parameters. The SIT parameter that controls this facility is the FILECTL parameter (ADDCICSSIT and CHGCICSSIT CL commands). The first element in the FILECTL parameter controls the maximum number of open files within each user shell. The second element indicates the amount of time a file can remain open within each user shell, without any activity against the file. For example, suppose that the FILECTL parameter were set to FILECTL(5 15). This would be interpreted as follows:

- The maximum number of CICS files that will be open at one time within the shell is 5.
- If a file has *not* had any activity in the last 15 minutes, the file is closed to the user shell.

The FILECTL elements work in parallel and also independently of each other. For example, setting the “maximum number of files” to 0 (zero) tells CICS that each

user shell can have an unlimited number of open files. However, if the second element, **No activity**, is set to 15, any file that has not been accessed in 15 minutes is closed.

It is the shell's physical files that are considered closed to the shell. The closed status of these files will not be apparent if a CEMT INQ or EXEC CICS INQUIRE command is issued, because this type of inquiry reflects a file's status for the control region as a whole; that is, the latest result of use by many shells. Automatic file closure is concerned with each shell and its data path to each shared file in the control region.

Any files opened to the shell, including those for temporary storage and transient data, are eligible for automatic closure.

When does automatic file closure occur?

Automatic file closure occurs at CICS task termination within each user shell. CICS checks its internal control blocks for open files, and closes files based on the maximum number allowed to be open and the least recently used (LRU) files. Any files, which are still open but which have not been used for a period exceeding that specified in the **No activity** setting, are closed. In either case, the LRU files are closed, using the criteria set in the FILECTL parameter.

Shared open data paths restrictions

CICS opens files without shared open data paths (ODP). If the file is created to allow shared ODPs or is overridden to allow shared ODPs before CICS opens the file, then when CICS attempts to open the file, an open error occurs. If the open request is on behalf of an EXEC CICS command from within an application program, the IOERR condition is set. If the open request occurs in the control region, the open fails and an error message indicating the problem is written to the control region job log.

Recoverable files

A file is defined to CICS/400 as being recoverable by specifying RECOVER(*YES) in the FCT definition. Note that this is the default setting for the RECOVER parameter. See "Using the ADDCICSFCT command" on page 177 for details.

Before a recoverable file can be used in a CICS system, the file:

- Must have been created
- Must have been registered to an OS/400 journal

The use of OS/400 commitment control is described in the File Management topic of the iSeries Information Center.

However, it is important to note that OS/400 commitment control puts further restrictions on CICS/400. These restrictions are detailed in "Recovery considerations" on page 104.

Example FCT definitions

The following are examples of FCT definitions, and the OS/400 methods for defining the objects referred to by these definitions. These examples use the simplest definitions to illustrate the relationship between FCT and physical file definitions. Default values are used wherever possible. For further information, refer to the CL topic in the iSeries Information Center.

Note: RECOVER(*YES) is the default on the ADDCICSFCT command. All the files defined in the examples need to be registered to an OS/400 journal using the STRJRNPFC command. See "Recoverable file commitment control requirements" on page 104 for further information.

KSDS examples

These examples show how to define a KSDS and two alternate indexes to that file. In all three examples, the DDS, the FCT definition, and the file creation command are given.

Figures 32 and 33 show the definitions required for KSDS EMPNO. The file needs a DDS and a physical file should be created using the CRTPF command.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	A*	11/17/92
200	A* KSDS - EMPNO file	11/17/92
300	A*	11/17/92
400	A* Employee Number File	11/17/92
500	A*	11/17/92
600	A	11/17/92
700	A R EMPNO	11/17/92
800	A DEPT 3A	11/17/92
900	A EMPNO 3A	11/17/92
1000	A NAME 30A	11/17/92
1100	A SSN 9A	11/17/92
1200	A K EMPNO	11/17/92

Figure 32. KSDS file EMPNO: data definition specification

Create physical file command	
CRTPF	FILE(MYLIB/EMPNO) SRCFILE(MYLIB/QDDSSRC) SRCMBR(EMPNO)
FCT entry	
ADDCICSFCT	LIB(MYLIB) GROUP(FCTS) FILEID(EMPNO) FILE(MYLIB/EMPNO) RCDACT(*ADD *BROWSE *NODLT *READ *UPD)

Figure 33. KSDS file EMPNO: create file command and FCT entry

Figures 34 and 35 define an alternate index for KSDS EMPNO. Note that a logical file should be created, using the CRTLF command, not a physical file.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	A*	11/17/92
200	A* KSDS - EMPDEPT file	11/17/92
300	A*	11/17/92
400	A* Employee Department File	11/17/92
500	A*	11/17/92
600	A R EMPDEPT	11/17/92
700	A DEPT 3A	11/17/92
800	A EMPNO 3A	11/17/92
900	A NAME 30A	11/17/92
1000	A SSN 9A	11/17/92
1100	A K DEPT	11/17/92
1200	A K EMPNO	11/17/92

Figure 34. Alternate index file EMPDEPT: data definition specification

Create logical file command

```
CRTLF FILE(MYLIB/EMPDEPT) SRCFILE(MYLIB/QDSSRC) SRCMBR(EMPDEPT)
FCT entry
ADDCICSFCT LIB(MYLIB) GROUP(FCTS) FILEID(EMPDEPT) FILE(MYLIB/EMPDEPT)
RCDACT(*ADD *BROWSE *NODLT *READ *UPD)
```

Figure 35. Alternate index file EMPDEPT: create file command and FCT entry

Figure 36 and Figure 37 defines another alternate index file for KSDS EMPNO. Note again that a logical file should be created using the CRTLF command.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	A*	11/17/92
200	A* Alternate index on the EMPNO file	11/17/92
300	A*	11/17/92
400	A* Employee Social Security Number File	11/17/92
500	A*	11/17/92
600	A R EMPSSN PFILE(MYLIB/EMPNO)	11/17/92
700	A EMPNO 3A TEXT('Employee Number')	11/17/92
800	A NAME 30A TEXT('Employee Name')	11/17/92
900	A SSN 9A TEXT('Social Security Number')	11/17/92
1000	A K SSN	11/17/92

Figure 36. Alternate index file EMPSSN: data definition specification

Create logical file command

```
CRTLF FILE(MYLIB/EMPSSN) SRCFILE(MYLIB/QDSSRC) SRCMBR(EMPSSN)
FCT entry
ADDCICSFCT LIB(MYLIB) GROUP(FCTS) FILEID(EMPSSN) FILE(MYLIB/EMPSSN)
RCDACT(*NOADD *BROWSE *NODLT *READ *NOUPD)
```

Figure 37. Alternate index file EMPSSN: create file command and FCT entry

ESDS examples

Figure 38 shows an example definition for an ESDS with fixed-length records. The record length is defined in the CRTPF command.

Create physical file command

```
CRTPF FILE(MYLIB/ESDSFIX) RCDLEN(100)
```

FCT entry

```
ADDCICSFCT LIB(MYLIB) GROUP(FCTS) FILEID(ESDSFIX) FILE(MYLIB/ESDSFIX)
ACCMTH(*ENTRY) RCDACT(*ADD *BROWSE *NODLT *READ *NOUPD)
```

Figure 38. Example commands for an ESDS with fixed-length records

Figures 39 and Figure 40 on page 93 show the creation of an ESDS with variable-length records. For this file, DDS is required to define a record containing a variable-length field.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	A*	11/17/92
200	A* Example creation of a variable length record file	11/17/92
300	A*	11/17/92
400	A R ESDSVAR	TEXT('ESDSVAR')
500	A DEPT 513A	TEXT('Record Entry')
600	A	VARLEN

Figure 39. Variable-length ESDS file

```

Create physical file command
CRTPF FILE(MYLIB/ESDSVAR) SRCFILE(MYLIB/QDSSRC) SRCMBR(ESDSVAR)
FCT entry
ADDCICSFCT LIB(MYLIB) GROUP(FCTS) FILEID(ESDSVAR) FILE(MYLIB/ESDSVAR)
ACCMTH(*ENTRY) RCDfmt(*VARIABLE)
RCDACT(*ADD *BROWSE *NODLT *READ *NOUPD)

```

Figure 40. Variable-length ESDS file: FCT entry and create file command

RRDS example

RRDS are fixed-length record files and are accessed by relative record number. Although DDS can be used to create the file, the DDS is not used by CICS/400 and therefore is not required.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	A*	11/17/92
200	A* Example creation of a relative record file	11/17/92
300	A*	11/17/92
400	A R RRDS	TEXT('RRDS file')
500	A RECENTRY 2000A	TEXT('Record Entry')

Figure 41. RRDS file: data definition specification

Figure 41 shows an example DDS for an RRDS file with a fixed record length of 2000 bytes. Before an RRDS file can be used by CICS/400, the number of records accessible to CICS/400 must be predefined.

If the file is to contain initial data, you can use the CPYF command to write this data to the file. The number of records copied to the file will be the number of records accessible to CICS/400. If the application is going to add further records to the end of the file, using the EXEC CICS WRITE command, you must use the initialize physical file member (INZPFM) command, with the parameter RECORDS(*DLT) and the TOTRCDS parameter specifying the maximum number of records that can be added to the file by CICS/400. Any attempt by a CICS application to add records beyond this limit will return an error condition.

If the file will not contain any initial data, then you should initialize the file using the INZPFM command with the RECORDS(*DLT) parameter. You can either specify the number of records in the TOTRCDS parameter or allow the parameter to default to the value *NEXTINCR.

In the example in Figure 42 on page 94, the file is created and initialized to have 10 000 records, the default number.

```

Create physical file command
CRTPF      FILE(MYLIB/RRDSFILE) RCDLEN(2000)
INZPFM    FILE(MYLIB/RRDSFILE) RECORDS(*DLT)
FCT entry
ADDICSFCT LIB(MYLIB) GROUP(FCTS) FILEID(RRDSFILE) FILE(MYLIB/RRDSFILE)
          RCDfmt(*FIXED) ACCMTH(*REL)
          RDCACR(*ADD *BROWSE *DLT *READ *UPD)

```

Figure 42. Example definitions for an RRDS

Interval control considerations

The CICS interval control program, in conjunction with a time-of-day machine instruction (MI) interface maintained by CICS, provides functions that can be performed at a specific time; such functions are called **time-controlled** functions. The primary task of the CICS interval control facility is the handling, synchronization, and initiation of tasks requested by user application programs and CICS internal service modules. Other functions also include obtaining and formatting time requests for the user.

Timer-related tasks

Interval control includes timer event driven tasks that initiate actions scheduled by CICS commands. This facility examines the Interval Control Element (ICE) storage chain contained in the CICS shared system storage for expired events, and initiates the appropriate action. After the ICE has ended, its entry is removed from the chain.

A timer-related task can be in any of three states:

- Unexpired—the expiration time for the task or event is still in the future
- Expired but suspended due to a CICS/400 resource being unavailable
- Expired and waiting to run

When a timer-related task expires, interval control attempts to schedule the task for execution if all the CICS resources that it requires are available. If any of the resources are not available, the entry is said to be suspended awaiting a resource to become available. When it is determined that all resources are available, the entry is then said to be enabled awaiting execution.

For more information about interval control, see the *CICS for iSeries Application Programming Guide*.

Protected interval control start requests

Protected interval control start requests are considered to be recoverable within a CICS/400 system. Interval control uses the TS/TD recoverable file to hold the information for protected interval control start requests, along with any data being passed to the starting task. The information related to protected interval control start requests is backed out during rollback.

You need to understand how CICS recoverable resources are registered to OS/400 journals and opened under OS/400 commitment control. Applications written to run under CICS/400 must conform to these rules. See “Recovery considerations” on page 104 for these restrictions.

Interval control elements

Interval control start requests are either protected or non-protected, depending on the START command options (see the *CICS for iSeries Application Programming Guide*), and are held in the appropriate TS/TD files AAEGxxxxTR and AAEGxxxxTN. This information is also restored during a warm or emergency start of a CICS control region, depending on the SIT parameters. Interval control uses shared storage in the control region (AEGxxxxSYS object) to hold CICS/400 areas pertinent to interval control requests. If the applications in the CICS system perform frequent interval control requests, be sure to allocate more space for control region space objects (SHRSTG parameter in ADDCICSSIT).

If an interval control request for storage fails because of lack of space, a series of AEG08xx messages is written to the control region job log. If this occurs, be sure to increase the size of storage before starting the control region again.

Acquiring terminals

Many interval control start requests are scheduled against CICS/400 terminals. If the OS/400 terminal device is in use outside CICS, interval control keeps trying to acquire the device to process the start requests. If a terminal-related interval control start does not begin at the requested time, the most likely cause is that the terminal cannot be acquired. The terminal must be in use by the CICS system before the start request can be scheduled against the terminal.

Interval control batch shells

Interval control uses special CICS shells, called **interval control batch shells**, to process interval control requests. The number allowed for a CICS system is determined by the ITVCTL parameter of the ADDCICSSIT command. You need to be sure that the number of these shells allocated for the control region is high enough to handle peak interval control processing loads in a timely manner. See “Defining the system initialization table (SIT)” on page 61 and “Using the ADDCICSSIT command” on page 239 for further information about the ITVCTL parameter.

Interval control start requests may not necessarily begin at the requested time. Interval control cannot control the scheduling within the OS/400 of the interval control batch shells allocated by the system administrator. Interval control start requests may *not* begin for a number of reasons, including:

- OS/400 job class for the interval control batch shells is inactive
- Security failure
- Interval control batch job failure

For timely processing of interval control start requests, you should ensure that the job queue for the interval control batch shells is set up to allow at least the minimum number of batch shells to be executed.

If there are not enough interval control batch shells allocated to handle all the outstanding requests, and you do not want to recycle the control region to change the number of interval control shells; you can use a special STRCICSUSR request to start an additional interval control shell. This request *must* be submitted in batch. The CL command is:

```
SBMJOB CMD(STRCICSUSR CTLRGN(xxxx) TRANID(*DATA)
        DATA(*CICS ICB))
```

Note: The spacing in the DATA parameter must be exactly as listed above with five blanks between *CICS and ICB.

Request identifiers

The interval control facility computes request identifiers for its own requests. Interval control uses the name of the temporary storage queue as the request identifier. Temporary storage is also used for the data associated with an interval control request.

Start requests and autoinstalled terminals

Interval control start requests against autoinstalled terminals are deleted during startup of a CICS control region. The autoinstalled terminal definition no longer exists in the TCT during the startup, therefore the start request cannot be scheduled against an undefined terminal.

To prevent this, the terminal associated with the start request should not be defined as an autoinstall terminal. Instead it should be defined as a terminal in the TCT; it should not default to using an autoinstall model.

Note: Client terminals are always autoinstalled and, therefore, interval control requests for client terminals are deleted during startup of a CICS control region.

Journal control considerations

CICS/400 journal control provides support for user journaling with the EXEC CICS WRITE JOURNALNUM command, in addition to that provided by automatic journaling of file operations specified through the FCT. In CICS/400, journal control is not used for recovery and restart processing, because OS/400 commitment control facilities already exist to provide these functions. CICS/400 user journals should therefore not be confused with OS/400 journals and journal receivers. OS/400 journals and journal receivers are an integral part of OS/400 commitment control, which is used by CICS/400 for recovery and syncpoint management. For more information on OS/400 journaling of CICS recoverable resources, see page 28.

User journals are special-purpose nonrecoverable sequential files that may be used as an audit trail, or as a change file of updates and additions, or for any other purpose needed by a CICS application program.

Journal records

User journals are defined to CICS in the journal control table (JCT). Data may be written to any journal specified in the JCT. The JCT may define one or more journals on direct access storage. Each journal is identified by a number known as the journal identifier. This number is in the range 1 through 99.

Note: In CICS/400, the journal identifier 1 is not the system log. It is a user journal and has the same attributes as journal identifiers 2 through 99.

Each journal record begins with a standard length field (LL), a user-specified identifier, and a system-supplied prefix. This data is followed in the journal record by any user-supplied prefix data (optional), and finally by the user-specified data.

CICS/400 user journals are defined as variable-length record physical files. To use these files in a batch application, you must know the physical file information for the journal you want, and the format of the CICS user journal records.

Journal output synchronization

If the requested journal file is defined as switchable, you do not have to create a physical journal file. The system will do this automatically. CICS switches to a new journal file and writes the journal record to the newly-created journal file automatically when the first journal file is full. The task requesting the write waits until the journal record is written to the new journal file. If the second file becomes full, another switch to a new journal file occurs. Journal switching occurs when necessary, unless CICS is unable to create the new journal file. Switchable journal files are created with the name AEGJ*Cnnxxx*, where *nn* is the journal ID and *xxx* is a number in the range 001 through 999 representing the generation of the switchable journal. The number is incremented by 1 each time a new switchable journal is created.

If a journal file is defined as nonswitchable, you will have to create a physical journal file. When the file becomes full, CICS closes the journal file, marks the JCT entry as CLOSED, and returns an IOERR condition to the application program.

Creating CICS/400 user journals

CICS/400 user journals are variable-length record, OS/400 arrival-sequence physical files. Because they are of variable length, they must be created using DDS, or by copying the CICS/400 journal file template provided with CICS/400. The journal file template is called QCICS/AAEGJCTMPT, and has a maximum record length of 32 742 bytes. See “Example creation of CICS/400 journal” on page 99.

If your user journal records are all significantly shorter than 32 742 bytes, you can avoid wasting DASD space by creating your own journal files using DDS. The application’s maximum record length is used as the journal file maximum record length.

Journal prefix area

Each CICS/400 journal record contains journal prefix information that can be used by application programs. Allow for this when creating the physical files used for CICS/400 user journaling. The contents of the journal prefix are shown in Table 18 on page 98.

User journal restriction

User journals cannot be registered to OS/400 journals. OS/400 does not open user journals under commitment control, therefore OS/400 journaling will not make the user journals recoverable.

Using non-switchable journals

When the record limit (MAXRCDS in CRTPF command) is reached and there are no more extents available for the file, CICS/400 indicates that the file is no longer available by closing the JCT entry.

Using switchable journals

For user journals defined as switchable in the JCT, when a journal switch occurs, CICS uses the journal file template QCICS/AAEGJCTMPT to create the new physical file. The journal record limit (RECLMT in ADDCICSJCT) determines when a journal switch occurs.

The physical file (on a switch) is created in the library specified in the journal library (JRNLIB in ADDCICSJCT). The file created is named AEGJCNmxxx, where *nm* is replaced by the journal file ID (JFILE in ADDCICSJCT) and *xxx* is replaced by a generation number, beginning with 001. The generation number is incremented each time a journal switch occurs.

Optionally, you may have a user job submitted when a journal switch occurs. CICS/400 provides a Local Data Area (LDA) that can be used by the user job for information about the journal file. The layout of the LDA is shown in Table 19.

Table 18. CICS/400 journal prefix area

Journal prefix area	Description	Format
Record length	Included only in open fixed record length; not included in open variable record length	Binary 4
Journal record length	Length of the journal record	Binary 4
Journal date	Date the journal record was written in EIBDATE format	Packed decimal 7
Journal time	Time the journal record was written in EIBTIME format	Packed decimal 7
Job name	OS/400 job name from which the record was written	Character 10
User ID	OS/400 user ID	Character 10
Journal number	CICS/400 journal identifier	Packed decimal 3
Journal function ID	Indicates the type of record being journaled	Character 2
File identifier	For automatic journaling of files. Indicates the file name in which the record resides	Character 8
Transaction identifier	CICS/400 transaction identifier	Character 4
Terminal Identifier	CICS/400 terminal identifier	Character 4
Journal user prefix offset	Offset from the Journal record length field, in the journal record where the user prefix area (if any) begins	Binary 4
Journal user prefix length	Length of the user prefix area (if any)	Binary 4
Journal record data offset	Offset from the Journal record length field, in the journal record where the data (if any) being journaled begins	Binary 4
Journal data length	Length of the journal data (if any)	Binary 4
Reserved	Reserved	Character 2
Relative byte address	The RBA of an ESDS record, or low values for a KSDS, RRDS, or API journal control write	Character 4

Table 19. Local data area layout

LDA name	Description	Format
Control region name	Control region name	Character 4
Date	Date in QDATFMT format	Packed decimal 7
Time	Time in QTIMFMT format	Packed decimal 7
Journal number	Journal number	Packed decimal 3
Library name	Library the physical file resides in	Character 10

Table 19. Local data area layout (continued)

LDA name	Description	Format
File name	File name of the physical file	Character 10
Member name	Member name in file; matches the file name	Character 10

Example creation of CICS/400 journal

Figure 43 shows the DDS source of the CICS/400 supplied user journal file template.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	A*	11/17/92
200	A* Switchable Journal File Template	
300	A R JOURNAL	TEXT('CICS/400 Journal Control Rec')
400	A RECENTRY 32740A	TEXT('Record Entry')
500	A	VARLEN

Figure 43. User journal file template: data definition specification

Figure 44 shows example definitions for creating a user journal called PAYROLL by copying the CICS/400 supplied template. The file specification is then changed to give the file an unlimited size.

Copy file command	
CPYF	FROMFILE(QCICS/AAEGJCTMPT) TOFILE(CICSWORK/PAYROLL) TOMBR(PAYROLL) MBROPT(*ADD) CRTFILE(*YES)
Change file command	
CHGPF	FILE(CICSWORK/PAYROLL) SIZE(*NOMAX)

Figure 44. Creating a user journal

Figure 45 shows how the user journal file template DDS source may be modified to change the maximum record length.

SEQNBR	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8	Date
100	A*	11/17/92
200	A* Example creation of a user journal	11/17/92
300	A*	11/17/92
400	A R USERJRN	TEXT('User Journal File')
500	A RECENTRY 200A	TEXT('Record Entry')
600	A	VARLEN

Figure 45. User journal file: data definition specification

Figure 46 shows how you can use the CRTPF command to create a user journal file.

Create journal file	
CRTPF	FILE(MYLIB/USERJRN) SCRFILE(MYLIB/QDSSRC) SRCMBR(USERJRN)

Figure 46. Creating a user journal using the CRTPF command

Example JCT definitions

Figure 47 shows an example JCT entry for journal file 60, which is disabled at control region startup. The journal will not be switched when full. The physical file to be used for the journal is called CICSWORK/PAYROLL.

```
ADDCICSJCT LIB(CICSWORK) GROUP(GROUP2) JFILE(60) JRNSTS(*DISABLED)
           JRNSWT(*NO) JRNFILE(PAYROLL)
```

Figure 47. Example JCT entry for a nonswitchable journal file

Figure 48 shows the JCT entry for journal file 50. This journal will switch at control region startup and after 50 000 records. Each new journal is created in library JOURNALLIB and the new journal name will be generated by the system. See “Using the ADDCICSJCT command” on page 202 for more information.

```
ADDCICSJCT LIB(CICSWORK) GROUP(GROUP1) JFILE(50) JRNLIB(JOURNALLIB)
           RECLMT(50000) NEWJRN(*YES)
```

Figure 48. Example JCT entry for a switchable journal file

Device considerations

This section describes the type of information you need to gather to define terminals, printers, and remote systems for your CICS system.

Defining local CICS terminals

Terminal control table (TCT) entries are required for any iSeries terminal devices that need to use, or are to be used by, a CICS/400 system. For CICS/400, a terminal may be a 5250 terminal, a 3270 terminal, or an SCS printer. By defining the devices in a terminal control table, you can control which terminals have access to a particular CICS/400 system.

You can restrict access to your system by defining TCT entries to specific OS/400 devices. If CICS access is to be allowed from only a few selected terminals in your OS/400 network, define an individual TCT entry for each. However, if you want to allow wider access to CICS without having to define each individual terminal in a separate TCT entry, you may choose to use the autoinstall capability of CICS. See Chapter 11, “Autoinstall for terminal definitions” on page 305 for details of the autoinstall capability.

Example TCT definitions

Figure 49 shows a TCT definition for terminal AUT1. This terminal may be used as both an autoinstall model for 5250 device types and as a terminal definition for iSeries device DEVICE1.

```
ADDCICSTCT LIB(MYLIB) GROUP(GROUP1) CICSDEV(AUT1) DEVTYPE(5250)
           DEVD(DEVICE1) DEVMODEL(*BOTH)
```

Figure 49. Example TCT entry for a 5250 terminal

Figure 50 shows a TCT definition for terminal AUT2. This definition can be used as an autoinstall model for 3270 device types and as a terminal definition for iSeries device DEVICE2.

```
ADDCICSTCT LIB(MYLIB) GROUP(GROUP1) CICSDEV(AUT2) DEVTYPE(3270)
           DEVD(DEVICE2) DEVMODEL(*BOTH)
```

Figure 50. Example TCT entry for an autoinstall terminal

Figure 51 shows an example TCT definition for a terminal TRM3. The terminal is disabled at control region startup. See “Using the ADDCICSTCT command” on page 275 for a description of all the parameters.

```
ADDCICSTCT LIB(CICSWORK) GROUP(ACCT) CICSDEV(TRM3) DEVTYPE(5250)
           DEVD(CICSTRM3) DEVSTS(*DISABLED) UNATTEND(*YES) UCTRN(*YES)
           VALIDATION(*YES) LIGHTPEN(*YES) SHIP(*YES) DEVACQ(*YES)
```

Figure 51. Example TCT entry for a 5250 terminal

Figure 52 shows an example TCT entry for 3270 terminal TRM4.

```
ADDCICSTCT LIB(CICSWORK) GROUP(ACCT) CICSDEV(TRM4) DEVTYPE(3270)
           DEVD(TERMINAL) USRARASIZE(25) DEVCHRID(6445) KATAKANA(*NO)
           SOSI(*YES) UNATTEND(*YES) UCTRN(*YES) ALTSCN(27X132)
           VALIDATION(*YES)
```

Figure 52. Example TCT entry for a 3270 terminal

See “Using the ADDCICSTCT command” on page 275 for a description of all the parameters.

Defining remote systems

Terminal control system table (TCS) entries are required for all remote systems with which your CICS system is to communicate. These systems may be other CICS systems or any system on your network with which you want to communicate using CICS Advanced Program to Program Communication (APPC) commands.

TCS entries typically have device definitions defined within the iSeries, which CICS uses to communicate with the remote systems. For more information about the iSeries network configurations that CICS requires, see *CICS for iSeries Intercommunication*.

In order for CICS/400 to be able to communicate properly with other CICS systems, you must ensure that the supplied transaction group INSGRP(AEGISC) INSLIB(QCICS) is defined as a group in the group list table for any control region with remote resource definitions.

Example TCS entry

Figure 53 on page 102 shows an example TCS definition for remote system SYS1. The network name is CICSSYS1. By default, the status at startup is enabled. Three send and three receive sessions are allowed.

```
ADDCICSTCS LIB(CICSWORK) GROUP(ACCT) SYSID(SYS1) NETWORK(CICSSYS1)
          SNDPFX(S1) SNDLMT(3) RCVPFX(I1) RCVLMT(3)
```

Figure 53. Example TCS entry

See “Using the ADDCICSTCS command” on page 261 for a description of all the parameters.

Remote resource considerations

It is possible for one CICS system to access any resource defined on another CICS system. This section describes the steps you need to take in order to access remote resources, and how to allow remote CICS terminals to access resources in your CICS system.

Defining remote resources

1. Ensure that your CICS system and the remote CICS system mutually define each other in their respective TCS tables, and that the relevant network configurations are in place.
2. Define the resource locally to your CICS system, using the appropriate resource definition command and specifying the SYSID that you gave the remote system in the TCS definition as the “Remote CICS system” on the resource definition.

Defining remote terminals

Remote terminals may be defined as described for remote resources. However, it is also possible to define local TCTs as being “shippable”. This permits the local terminal definition to be shipped to the remote system when and if required, without having to have an explicit TCT definition for the terminal on the remote system.

Example TCT definitions for remote and shippable terminals

Figure 54 shows an example definition for terminal SUR1.

```
ADDCICSTCT LIB(CICSWORK) GROUP(ACCT) CICSDEV(SUR1) DEVTYPE(3270)
          USRARASIZE(25) DEVMODEL(*TERMINAL)
          SYSID(remote sysid) RMTDEV(remote terminal ID)
```

Figure 54. Example TCT entry for a remote terminal

Figure 55 shows an example definition for terminal SHP1.

```
ADDCICSTCT LIB(CICSWORK) GROUP(ACCT) CICSDEV(SHP1) DEVTYPE(3270)
          USRARASIZE(25) DEVMODEL(*TERMINAL)
          SHIP(*YES)
```

Figure 55. Example TCT entry for a shippable terminal

Printer spooling considerations

Applications using printer spooling commands must create a printer file for each file to be opened. The files are determined by the CLASS option of the EXEC CICS SPOOLOPEN command. The CLASS defaults to “A”. Figure 56 on page 103 shows a template for an EXEC CICS SPOOLOPEN command and the corresponding

CRTPRTF CL command needed to create the printer file. The example commands create print spooling files for classes A and P. The control region ID is SAMP.

Class A Spool to spool file, no carriage control, no maximum number of records (*NOMAX).

Class P Direct to printer PRT01 with ASA carriage control.

```
EXEC CICS SPOOLOPEN command for class A
EXEC CICS SPOOLOPEN
...
END-EXEC.

Create print file command
CLASS_A:  CRTPRTF  FILE(MYLIB/AEGSAMPPA) CTLCHAR(*NONE)
          MAXRCDS(*NOMAX)

EXEC CICS SPOOLOPEN command for class P
EXEC CICS SPOOLOPEN
          CLASS("P")
...
END-EXEC.

Create print file command
CLASS_P:  CRTPRTF  FILE(MYLIB/AEGSAMPPP) DEV(PRT01) CTLCHAR(*FCFC)
          SPOOL(*NO)
```

Figure 56. Example spool file creation commands

The printer files for spooling must be created using the CRTPRTF command and must meet the following requirements:

- The printer file must be created in a library that exists in the library list for the user executing CICS/400 printer spooling commands.
- The files must be created with a file name of AEGxxxxPn, where xxxx is replaced by the CTLRGN from which these files are to be accessed, and n is replaced by the value for the CLASS option of the EXEC CICS SPOOLOPEN command. This value may be any character allowed in the FILE parameter the CRTPRTF CL command.
- The printer file may either be defined as a spool file or go directly to the printer. This may depend on the individual use of the file for a specific user, or on your own installation standards. However, if you intend to do SPOOLWRITES with a data-area that exceeds the width of the printer-file page, then you should define the print file with option SPOOL(*YES). If you do not, CICS/400 cannot determine whether to fold the data stream over multiple lines.
- If direct printing is required, you must first stop the print writer using the CL command ENDWTR.
- The remainder of the printer file definition depends on the use of the printer file by the application. The application developer should supply any additional information necessary to define the printer file properly for an application.

Printer spool files are always released at task syncpoint time.

The Override with Printer File (OVRPRTF) CL command affects the way the EXEC CICS SPOOLOPEN command options are handled. The SECURE(*YES) parameter of the OVRPRTF CL command prevents any further overrides of the printer spool file, and inhibits the EXEC CICS SPOOLOPEN command options. The

CTLCHAR(*NONE) parameter of the OVRPRTF CL command inhibits the ASA option on the EXEC CICS SPOOLOPEN command. Prints are not formatted, and the ASA control character is treated as a printable character.

Trace considerations

At CICS startup you determine whether CICS internal trace is to be active, and set the size of the internal trace table. In addition, CICS auxiliary trace and user trace status are determined at startup using the trace parameters in the SIT. The trace user space objects named in the SIT are created at control region startup if they do not already exist.

Note: The size chosen for the internal trace table also determines the size of each of the auxiliary trace user space objects.

The maximum size of each CICS trace record is approximately 4KB.

Internal trace

At control region startup, CICS/400 creates an internal trace space object called QTEMP/TRACETABLE. The calculation used to determine the size of the space object is:

Number of table entries X maximum size of a CICS/400 trace entry.

Printing CICS/400 trace

The PRTCICSTRC command can be used to format and print CICS/400 trace entries recorded in the auxiliary trace objects. This command is described in detail in *CICS for iSeries Problem Determination*. An example trace print and use of trace is described in *CICS for iSeries Problem Determination*.

Recovery considerations

There are a number of CICS/400 resources that may be defined as recoverable, namely CICS/400 files, intrapartition transient data queues, and auxiliary temporary storage queues. Protected interval control starts are always recoverable.

For a CICS/400 resource to be recoverable, it should be:

- Defined as recoverable in the appropriate resource definition table
- Registered to an OS/400 journal and journal receiver

OS/400 journals and journal receivers should not be confused with CICS/400 user journals. OS/400 journals and journal receivers are an integral part of OS/400 commitment control, which is used by CICS/400 for recovery and syncpoint management. Each recoverable resource is opened within CICS under the commitment control facilities of OS/400.

Recoverable file commitment control requirements

CICS/400 uses the commitment control facilities of the OS/400 for all recoverable files, including the TS/TD recoverable file AAEGxxxxTR (see “Defining temporary storage and transient data files” on page 63). A dedicated OS/400 journal is used to coordinate resource recovery between local and remote CICS resources.

A number of command examples are provided to serve as guidelines. Refer to the CL topic in the iSeries Information Center for more details about any specific command. You need to use the command options required for your installation.

To enable a file to use OS/400 commitment control, three steps must be completed:

1. Using the CRTJRNRCV command, create an OS/400 journal receiver.
2. Using the CRTJRN command, create an OS/400 journal that uses this journal receiver.
3. Using the STRJRNPF, register the physical file to the OS/400 journal.

Failure to complete these steps for the TS/TD recoverable file AAEGxxxxTR causes control region startup to fail. Figure 57 shows an example of how to register the recoverable file to an OS/400 journal. The example program does not include any error checking or message monitoring.

```
Create journal receiver
CRTJRNRCV  JRNRCV(MYLIB/ACCTJRNRCV)
           TEXT('CICS/400 Journal receiver for ACCT control region')

Create a journal
CRTJRN  JRN(MYLIB/ACCTJRN) JRNRCV(MYLIB/ACCTJRNRCV)
        TEXT('OS/400 journal for CICS/400 journal receiver ACCT')

Register the file to the journal
STRJRNPF FILE(MYLIB/AAEGSAMPTR) JRN(MYLIB/ACCTJRN) IMAGES(*BOTH)
          OMTJRNE(*NONE) /* 'OS/400 journaling for control region -
          TS/TD recoverable file AAEGSAMPTR' */
```

Figure 57. Registering a recoverable file to an OS/400 journal

When a control region starts, CICS/400 syncpoint control requires an OS/400 journal of its own to use for coordinating recovery between CICS systems. This journal, its receivers, and the library they reside in, are created by the control region manager, (unless they already exist). The journal is named QRCVYJ and resides in a library called QCICS.ssss, where ssss is the four-character control-region identifier. The receivers reside in the same library and are called QRCVYRnnnn, where nnnn is a four-digit decimal number. You should not associate any other resources with these journals. You should not erase the objects (library, journal, or receivers) unless you are sure that the control region has no recovery information in them, and you are recommended not to use the libraries for any other objects.

OS/400 commitment control and logical unit of work (LUW)

OS/400 commitment control uses its journals to record changes to resources associated with them. A collection of reversible changes is accumulated until a syncpoint is taken. The collection of changes accumulated between consecutive syncpoints is called a logical unit of work (LUW). At syncpoint, an LUW is either committed, that is the changes are made permanent, or rolled back, that is the changes are removed. The changes for an LUW may be distributed among multiple journals or even across multiple OS/400 systems.

It may be that an LUW cannot be either committed or rolled back immediately a second syncpoint occurs. One reason for this is that remote systems may become unavailable just at the time when a decision about an LUW is being made. In this case the LUW becomes “in-doubt” until the systems can establish the conversation

again. While an LUW is in doubt, the resources involved in the changes may need to be locked. If the local OS/400 system is re-IPLed, the LUW resources may need to be re-locked, and so on.

In these and other situations, OS/400 commitment control and CICS/400 cooperate to preserve the integrity of CICS/400 recoverable resources.

All CICS/400 user shells run under commitment control. Commitment control is explicitly started by the shell (using the STRCMTCTL command) before any transactions are run, unless it has already been started. If a CICS/400 user shell started commitment control itself, it will also attempt to end it (using the ENDCMTCTL) when the shell ends.

If commitment control is started normally by CICS/400 but cannot terminate normally at the end of the shell, an abend occurs. There is one exception to this—if an external resource manager (for example, SQL) has registered an exit on commitment control and has failed to remove it, CICS/400 removes its own exit, issues a warning message, and continues to shut down. In this exceptional case, commitment control remains active after shell shutdown.

Chapter 8. Administering the control region

The concept of the control region was introduced in Chapter 1, “Introducing CICS® for iSeries™” on page 3; Chapter 5, “Defining a basic control region” on page 61 describes how the IVP control region was defined and Chapter 6, “Security requirements for CICS/400” on page 75 describes how to set up your own control region. This chapter provides more detail about how the control region manages the CICS/400 workload, and tells you how to start and stop a control region.

This chapter covers the following:

- “Introduction to the control region”
- “Control region processing” on page 108
 - “Control region initialization” on page 108
 - “Control region runtime processing” on page 109
 - “Control region shutdown” on page 111
- “Starting a control region (STRCICS)” on page 112
- “Ending a control region (ENDCICS)” on page 118

Inbound ISC processing assumes that the name of the subsystem matches the control region name. Evoked or prestarted jobs should run in this subsystem, but there is no requirement to run the control region in that subsystem. However, it is recommended that you run your control region in the subsystem with the same name. It is also recommended that you make the control region an autostart job entry, and set up job descriptions and job queues to run interval control batch shells in the subsystem. This ensures that, when the subsystem ends, all batch work associated with the control region also ends. For more information about ISC processing, see *CICS for iSeries Intercommunication*.

Introduction to the control region

The control region provides for the control, scheduling and work management mechanisms necessary to coordinate all the shared resources of a CICS environment:

- Shared system and user storage areas are initialized, managed, and released.
- Runtime resource definitions are built and validated.
- Resource definitions from the previous release are converted, as permitted by the STRCICS CONVERT parameter.
- Resource status is initialized or reset depending on startup options.
- Cleanup of shared resources from previous control region execution is provided when appropriate.

The control region uses queues to communicate with shells.

Control region initialization builds the shared resource environment common to all CICS jobs running under the same control region name (in OS/400 terms) or system identifier (in CICS terms). The initialization tasks, which are performed once per system and which affect all users, are:

- Making shared system code available
- Acquiring shared storage areas
- Loading CICS resource tables

- Building shared control blocks (for example, Common Work Area)
- Establish entry points for CICS

Control region processing

This section contains a more detailed description of the control region processes. Besides providing an overall understanding, this section could be useful in tracking down an error condition.

The CICS control region provides access to common CICS resources. The control region comprises three major processes:

- Control region initialization
- Control region runtime processing
- Control region shutdown

Note: An unexpected return code during any of these processes displays the following message:

```
AE61502 The CICS/400 control region &2 detected a serious error
```

where &2 is the name of the control region. A dump of the control region is taken, and a controlled shutdown starts. If the error occurs during resource definition, message AEG1530 is sent to QSYSOPR, asking whether you want to continue or cancel the operation.

Control region initialization

The initialization process is as follows:

- CICS verifies that the specified or default control region is not already operational. If it already exists, the message AEG1531 is displayed, asking whether the new control region should continue or abandon initialization.
- To provide addressability for the user shells to shared control region areas, an OS/400 user space is created in QGPL.
- The control region data queue is created in QTEMP.

Note: The data queue is the mechanism used to inform the control region that there is work for it to perform from one of the interactive or batch shells.

- The OS/400 job name, user ID, job number, job type, and job subtype of the control region are registered in internal control blocks.
- If resources from a previous release of CICS/400 are found, a message is sent to QSYSOPR. Whether or not the resources are converted to the current release, depends on the setting of the CONVERT parameter of the STRCICS command.
- A journal for syncpoint control is identified (or created) in the library QCICS.ssss, where ssss is the four-character control-region identifier. If a journal is found, it is scanned to determine whether there are any logical units of work that are yet to be resolved. If there are any, a *COLD startup is rejected, but any other form of startup may be allowed, depending on the SIT parameters. (See "Setting the STRTYPE parameter" on page 114.) The unresolved LUWs are recorded and their resources locked in the control region itself to allow safe resolution at a later time. See page 105 for more information about this journal.
- Various components within CICS need to perform special initialization of CICS shared tables (for example, TST, TCT, and internal trace). Those components requiring startup initialization procedures each have an entry (not modifiable by

the customer) in the control region initialization and termination header file. These entries define the component initialization programs for the control region program to invoke at startup.

- Storage services are called first to set up user spaces and system control blocks.
- Trace is initialized, if requested, to track problems that might occur.
- The resource management initialization routine ensures the consistency and validity of SIT values and loads initial values for the other resource tables.
- Transient data and temporary storage areas are built or rebuilt (*WARM or *EMER startup).
- Intersystem communication terminal entries for remote CICS sessions are installed.

ISC note: Inbound enabled links may have either OS/400 prestarted batch shells or evoked jobs started by the OS/400 STRPJ command. The limits on prestarted jobs should be set high enough to allow all the sessions (TCS entries) to be in use at the same time. Outbound enabled links are marked available. For further information, see *CICS for iSeries Intercommunication*.

- When the definition of the control region indicates that CICS Clients are to connect with TCP/IP, the TCP/IP listener jobs are started when the control region is initialized. These jobs are used to transmit the TCP/IP requests to CICS Intersystem Communication jobs over an internal APPC connection. As part of their initialization and operation, the TCP/IP listener job creates and uses APPC Controller Descriptions and Device Descriptions.
- The return code is checked before running the next module in sequence. If the return code indicates an error control region initialization is abandoned (message AEG1533 is issued). If the return code indicates a warning, the system operator is given the choice (AEG1530) either to allow the control region startup to continue disregarding the warning error, or to end the control region to correct the error.

If the operator cancels the startup, message AEG1532 is issued, and the control region abandons initialization and terminates.

When all control region startup processes have been completed with no serious errors, the system operator is sent a message (AEG1534) indicating that the control region is initialized and operational. CICS users are then allowed to access CICS transactions through use of the shell facility. The shell must be initiated with the same sysid as the control region under which it is to run.

Control region runtime processing

After initialization is complete, the control region waits for shell service requests, shell status updates, or timer driven processes.

- Shell service requests include resource status changes, extrapartition transient data requests, completion of ISC logical units of work (LUW), and shell status changes.
- Timer events include expiration of interval control start request times, and periodic housekeeping requests.

The control region processing loop, in which the control region receives work to do from CICS shells, can be described in the following steps:

1. CICS calls terminal control to wait for data queue records. The control region does not have any display or communication files open. It receives input only by data queue messages.
 - When a data queue message is returned:

If this is a request either to change or delete a shell dispatch control area, the appropriate action is performed by the storage services facility. A change request updates the status and shutdown fields. A delete removes the shell from this control region.

If the request can be handled immediately, the CICS service routine is called.

If the request cannot be handled when first requested (waiting for resources), it is deferred and handled as a deferred work element (DWE).
 - Timer driven events are processed.
2. The control region performs a scan of the task-termination DWEs for work ready to process and calls any specified functions in a manner similar to the data queue processing. If an internal call to a service module fails when attempting to process this work, message AEG1505 is issued.

Note: The DWE is the catalyst used by CICS for invoking “event-driven” processes. DWEs define processing to be performed during a predefined CICS event (for example, syncpoint or task termination).
3. The control region checks whether CICS is in shutdown mode. If not, the runtime processing loop is repeated until shutdown is indicated.

Handling processing requests

The following features govern the way processing requests are handled in the control region:

- OS/400 data queues created in the QTEMP library are used to communicate with the control region.
- Requests for control region services are sent and answered through the OS/400 data queues.
- Shared CICS resources are used and managed through the use of shared memory control blocks within the control region. All shared resources are governed by the control region or allocated by the control region at initialization.
- Only CICS service modules run within the control region. CICS/400 application programs run within the shell environment.
- The separation of duties of the control region and the shell facility, along with the separation of shared and nonshared storage, provides protection to the overall CICS/400 system.

Communication between control region and shell

The control region and user shells exchange information with each other. The control region ID identifies the control region with its associated user shells. Similarly, when a user shell is started by using the STRCICSUSR command, the CTLRGN parameter provides the link for the exchange of information with the identified control region.

Control region objects

These separate jobs must exchange information with each other. Many of the mechanisms for information exchange involve the creation of OS/400 objects with architected CICS names beginning with the “AEG” prefix. For example:

- OS/400 data queues QTEMP/AEGCICS *DTAQ created at control region and shell startup.

- OS/400 user space objects created in QTEMP at control region startup. These objects contain the directory of control region shared storage areas, CICS internal control blocks, and storage used only by CICS internal processes within the control region.
- An OS/400 user space QGPL/AEGxxxx created at control region startup, where the xxxx is the control region ID. Each shell requiring information exchange with a control region must be able to access this object.

Control region shutdown

The following commands or errors can cause shutdown to begin:

- The system administrator entered the CEMT PERFORM SHUTDOWN command on a terminal that had an active shell attached.
- The system administrator entered an ENDSBS CL command. for the subsystem in which the control region is currently running.
- The system administrator entered an ENDCICS CL command.
- A critical internal error.
- An ENDJOB command was issued.
- An OS/400 failure.
- An ENDSYS command was issued. This command ends all subsystems on the OS/400.

When shutdown has begun, the OS/400 system operator is notified. No new work for this control region may be started. Shell startup requests are rejected. The batch and user shells are notified by data queue messages of the control region shutdown and the type of shutdown.

The processes invoked for shutdown are described as follows:

- A data queue message is sent to the shell to trigger shutdown.
- If this is a controlled shutdown (no time limit), or a controlled shutdown whose delay time limit has not been met, the control region continues to wait for the shells to terminate normally. Shells are terminated if they are idle, when a transaction finishes, when CESF is entered, or when an ENDCICSUSR command is issued.

As each shell terminates, a data queue shutdown message is returned to the control region. When all shells associated with a control region have ended or the shutdown status has been changed to immediate, then shutdown proceeds.

- If you use the *IMMED parameter on the ENDCICS command, each shell connected to the control region is sent an immediate shutdown data queue message. Then the control region shuts down.
- The initialization/termination table for the control region is used to invoke modules used to clean up resources and user spaces within the control region. After each module is called, the return code is examined. Anything with a severity level of 20 or above causes message AEG1693 to be issued, a dump to be taken, and the control region to be brought down.
- The control region's user space is deleted.

On completion of shutdown, the system operator is sent a message indicating that the control region is terminated.

ISC note: If the ISC initialization was performed at control region startup (by issuing a Start Prestart Job (STRPJ) command), an End Prestart Job (ENDPJ) command to shut down the ISC shells should be issued. The

ENDPJ command is used instead of a data queue message because a prestarted job does not have a CICS data queue attached to it when the ENDCICS CL command is issued. The prestarted jobs are ended automatically by the ENDSBS command.

Subsystem note: If the CICS control region is running under a separate subsystem, the OS/400 ENDSBS command may be issued by an authorized operator. The ENDSBS command specifies *CNTRLD or *IMMED with an additional parameter to convert *CNTRLD into *IMMED after a specified delay time.

Starting a control region (STRCICS)

You should use the STRCICS CL command to start a control region and, optionally, to convert any resource definition files.

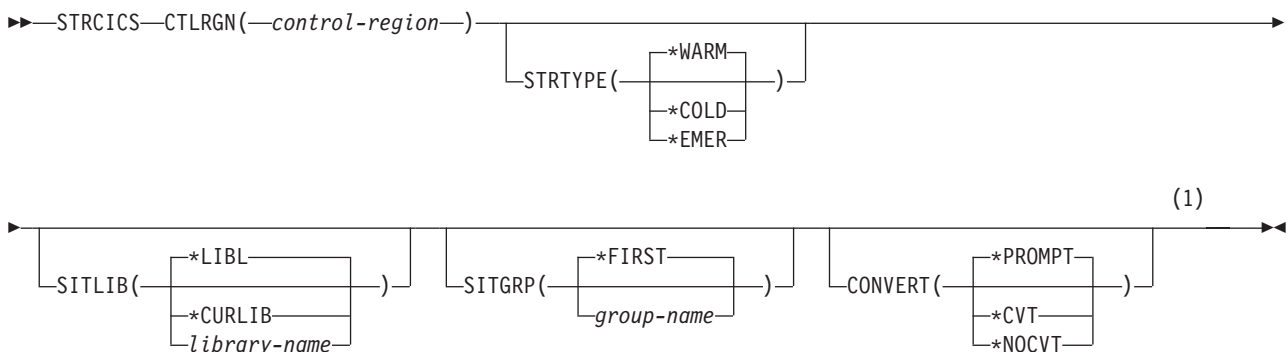
Note: If you are migrating to a new release of CICS/400, you should convert all your existing resource definition files using the INZCICS CL command as part of the IVP procedure. See "Migration" on page 21 for details.

CL command defaults

The defaults given in the CL command description are those that are supplied with the OS/400 system. You should check that your installation has not made any changes to these command default parameters.

STRCICS

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

The Start CICS (STRCICS) CL command starts a CICS control region and specifies how to recover CICS temporary storage and transient data files.

This command causes a *WARM start (which is the default) of the specified control region. The library list (*LIBL is the default value) is used to locate the CICS group that contains the CICS system initialization table (SIT) that specifies the values needed to begin control region initialization. For details on setting up a SIT, see

“Defining the system initialization table (SIT)” on page 61. The CICS control region must have completed its startup process before any associated shells (batch or user) can be started. Refer to the STRCICSUSR CL command (“Starting a user shell (STRCICSUSR)” on page 125) to start a CICS user shell.

See “Interpreting the syntax diagrams” on page 133 for an explanation of the conventions of syntax diagrams.

Required parameters

CTLRGN

The name of the CICS control region to be started, also known as the CICS system ID.

control-region: The name of the control region may be up to four characters in length. The first character must be alphabetic, or one of the special characters \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

You should not give the control region the name CICS.

Optional parameters

STRTYPE

indicates how to recover the CICS temporary storage and transient data files when the CICS control region is started. STRTYPE is needed only if you need to do a *COLD or *EMER start. The possible STRTYPE values are:

- ***WARM:**
- ***COLD:**
- ***EMER:**

The results of setting these values are described in “Setting the STRTYPE parameter” on page 114.

SITLIB

The name of the OS/400 library containing the group (SITGRP) that contains the CICS system initialization table to be used to initialize the CICS control region. This parameter is used if you need to bring in a different SIT for special processing.

The possible SITLIB values are:

***LIBL:** The library list is used to locate the first CICS group that contains the CICS system initialization table.

***CURLIB:** The current library for the job is used to locate the group that contains the system initialization table.

library-name: Specifies the name of the library used to locate the group that contains the system initialization table.

SITGRP

The name of the CICS group that contains the CICS system initialization table to be used to initialize the CICS control region.

***FIRST** No group is specified. The first group found is used.

group-name: The maximum length is ten characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CONVERT

indicates what happens to previous-release resource definition tables that are encountered during control region initialization.

STRCICS

The possible CONVERT values are:

***PROMPT:** Tables from previous releases of CICS require the operator to respond as to whether they get converted to the current release.

***CVT:** Tables from previous releases of CICS/400 are automatically converted by INZCICS to the current release.

All groups contained in the library are converted.

***NOCVT:** Tables from previous releases of CICS are not automatically converted to the current release.

When the system initialization table is from a previous release of CICS/400, then the control region is not started. Any other tables from previous releases allow the CICS control region to be started, but the data in the tables is ignored.

Examples

The STRCICS command starts a CICS control region called "TEST". The CICS control region is started with all CICS temporary storage and transient data files being cleared. It uses the CICS system initialization table located in the OS/400 library MYLIB and the CICS group MYGRP. If any resource definition files from the previous release are found, you will be asked whether or not you want these files to be converted. If you ask for the files to be converted, all files in the library will be converted.

```
STRCICS CTLRGN(TEST) STRTYPE(*COLD)
        SITLIB(MYLIB) SITGRP(MYGRP) CONVERT(*PROMPT)
```

Setting the STRTYPE parameter

Terminology

The term *recovery* is applied to both recoverable and nonrecoverable queues with the following meanings:

To recover a **nonrecoverable queue** means to leave it as it was at the preceding control region termination.

To recover a **recoverable queue unaffected by in-doubt units of work** means to make it as it was at the preceding control region termination, but with all the uncommitted changes of in-flight units of work removed.

To recover a **temporary storage recoverable queue affected by in-doubt units of work** means to restore it to the condition at control region termination. It can be read by any transaction but cannot be written until the in-doubt units of work are resolved.

To recover a **transient data recoverable queue affected by in-doubt units of work** results in the actions shown in Table 20.

Table 20. Recovery actions for TD queues affected by in-doubt units of work

Reader in-doubt Writer in-flight	Freeze read end. Roll back write end.
Reader in-doubt No writer	Freeze read end.
Reader in-doubt Writer in-doubt	Freeze both ends.
Reader in-flight Writer in-doubt	Restore read records. Freeze write end.

Table 20. Recovery actions for TD queues affected by in-doubt units of work (continued)

No reader Writer in-doubt	Freeze write end.
Note: "Freeze" means that the relevant end of the queue cannot be read from or written to until the in-doubt units of work in that area have been resolved.	

(A temporary storage queue can be affected by at most one in-doubt unit of work. A transient data queue can be affected by up to two in-doubt units of work, one as reader, and one as writer.)

STRTYPE values affect the recovery of the CICS temporary storage (TS) and transient data (TD) files. The way these files are recovered also depends on the TSCTL and TDCTL data recovery options in the CICS system initialization table. See "Using the ADDCICSSIT command" on page 239 for further information on setting these parameters. The recovery of these files also depends on whether there are in-doubt units of work (explained below) affecting them.

Operation with no in-doubt units of work

STRTYPE set to *WARM: When *NO is specified on the TSCTL data recovery option, the CICS temporary storage queues are cleared. When *NO is specified on the TDCTL data recovery option the transient data queues are cleared.

When *YES is specified on the TSCTL data recovery option, all recoverable and nonrecoverable temporary storage queues are recovered. When *YES is specified on the TDCTL data recovery option, all recoverable and nonrecoverable transient data queues are recovered.

STRTYPE set to *COLD: The CICS temporary storage and transient data queues are cleared.

STRTYPE set to *EMER: When *NO is specified on the TSCTL data recovery option, the CICS temporary storage queues are cleared. When *NO is specified on the TDCTL data recovery option, the transient data queues are cleared.

When *YES is specified on the TSCTL data recovery option, recoverable temporary storage queues are recovered (except those that are older than the specified age limit in the TSCTL Age limit element). When *YES is specified on the TDCTL data recovery option, all recoverable transient data queues are recovered.

Operation with in-doubt units of work

The use of two-phase commit protocols for data integrity in distributed processing makes it possible for a CICS or OS/400 failure to leave uncommitted changes to resources. (Without two-phase commit, these resources are returned to the state they were in before any changes were made, irrespective of how related changes in other processors are treated.)

With two-phase commit, if a system failure occurs in the middle of coordinating changes with a remote system, the remote system waits until the failing system recovers, and advises it whether changes in the failing processor are to be committed or backed out. If a unit of work originating in a CICS system is still waiting to hear from the remote system at the time the CICS control region is initialized, the unit of work is said to be **in-doubt**. In this case, any CICS resources in which it has uncommitted changes must be preserved until the unit of work is resolved.

STRCICS

This means that the CICS control region is not initialized, because this would destroy the integrity of the in-doubt units of work. This happens when the STRCICS command is used to start a CICS control region for which:

- There are in-doubt units of work waiting to be resolved,
- And, the STRTYPE parameter is *WARM or *EMER,
- And, an in-doubt unit of work has uncommitted changes to a recoverable TD/TS queue, and the data recovery option of TDCTL (or TSCTL depending on the queue) is *NO.

A message to say that this has happened will be sent to the job log. You are recommended to make the recovery option *YES, or to use OS/400 facilities to remove the in-doubt units of work. If there are no in-doubt units of work, the control region will be initialized as required by the STRTYPE and TDCTL parameters.

TS queue with age-limit: The situation is different when a TS queue has the age-limit option of TSCTL set. If the STRCICS command is used to start a CICS control region for which:

- There are in-doubt units of work waiting to be resolved,
- And, the STRTYPE parameter is *WARM or *EMER,
- And, an in-doubt unit of work has uncommitted changes to a recoverable TS queue,
- And the aging option of TSCTL requires such a queue to be deleted,

the queue will not be deleted, since this would destroy the integrity of the in-doubt units of work. If there are no in-doubt units of work, the TS queues will be discarded in accordance with the aging option of the TSCTL parameter.

The complete summary of recovery procedures for TD and TS queues (whether in-doubt units of work are involved or not) is summarized in the following sections.

CICS/400 recovery with a cold start

With a cold start, all recoverable and nonrecoverable queues are cleared. The settings of TSCTL and TDCTL have no effect. If there are in-doubt units of work waiting to be resolved, the control region is not initialized and a message is written to the job log. You should use a warm start or OS/400 facilities to remove the in-doubt units of work.

CICS/400 recovery with a warm start

The recovery actions are summarized in Table 21.

Table 21. CICS/400 recovery with STRTYPE=*WARM. In each row, the last entry is the action taken, as a result of the conditions in the preceding columns.

TSCTL recovery	TDCTL recovery	Action taken
*NO, but in-doubt	N/A	Abandon initialization.
*NO, not in-doubt	*NO, but in-doubt	Abandon initialization.
	*NO, not in-doubt	Clear nonrecoverable and recoverable queues.
	*YES	Reinstate nonrecoverable TD queues and discard nonrecoverable TS queues. Reinstate all recoverable TD queues, and discard the recoverable TS queues.

Table 21. CICS/400 recovery with STRTYPE=*WARM (continued). In each row, the last entry is the action taken, as a result of the conditions in the preceding columns.

TSCTL recovery	TDCTL recovery	Action taken
*YES	*NO, but in-doubt	Abandon initialization.
	*NO, not in-doubt	Reinstate nonrecoverable TS queues and discard nonrecoverable TD queues. Reinstate all recoverable TS queues, but discard recoverable TD queues.
	*YES	Reinstate nonrecoverable and recoverable queues.
Key: N/A Not applicable TD/TS Transient data/temporary storage In-doubt Control region terminated while containing in-doubt units of work with uncommitted changes to the relevant queues		

CICS/400 recovery with an emergency start

The recovery actions are summarized in Table 22.

Table 22. CICS/400 recovery with STRTYPE=*EMER. In each row, the last entry is the action taken, as a result of the conditions in the preceding columns.

TSCTL recovery	TDCTL recovery	TS age limit	Action taken
*NO, but in-doubt	N/A	N/A	Abandon initialization.
*NO, not in-doubt	*NO, but in-doubt	N/A	Abandon initialization.
	*NO, not in-doubt	N/A	Clear nonrecoverable and recoverable queues.
	*YES	N/A	Clear nonrecoverable queues. Reinstate recoverable TD queues and discard recoverable TS queues.
*YES	*NO, but in-doubt	N/A	Abandon initialization.
	*NO, not in-doubt	TS age limit	Clear nonrecoverable queues. For recoverable TS queues: <ul style="list-style-type: none"> • Reinstate under-age TS queues • Reinstate infected over-age TS queues • Discard the uninfected over-age TS queues Discard the recoverable TD queues.
		No TS age limit	Clear nonrecoverable queues. For recoverable queues, reinstate TS queues, but discard the TD queues.
	*YES	TS age limit	Clear nonrecoverable queues. For recoverable queues: <ul style="list-style-type: none"> • Reinstate TD queues • Reinstate under-age TS queues • Reinstate infected over-age TS queues • Discard the uninfected over-age TS queues
		No TS age limit	Clear nonrecoverable queues. Reinstate all recoverable queues.
	Key: N/A Not applicable TD/TS Transient data/temporary storage In-doubt CICS/400 stopped containing in-doubt units of work with uncommitted changes to the relevant queues Age limit The TSCTL parameter of CHGCICSSIT defines an age limit for TS data (so that "under-age" queues are recovered, but over-age queues are not) Infected Containing uncommitted changes from in-doubt units of work		

Ending a control region (ENDCICS)

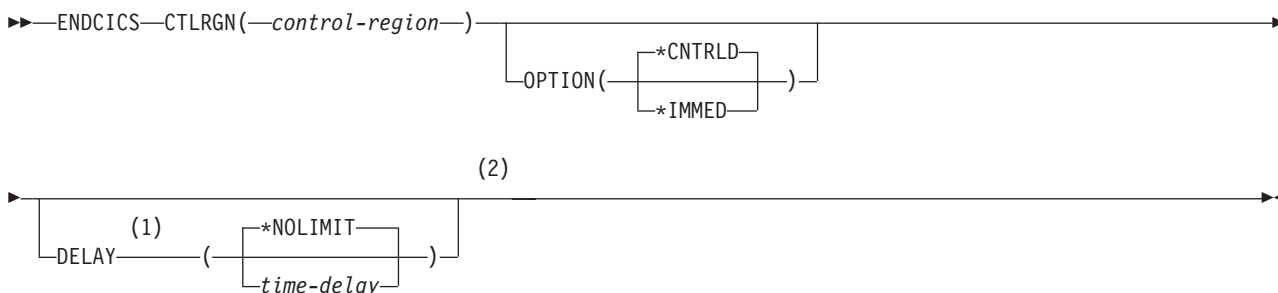
CL command defaults

The defaults given in the CL command description are those that are supplied with the OS/400 system. You should check that your installation has not made any changes to these command default parameters.

See "Interpreting the syntax diagrams" on page 133 for an explanation of the conventions of syntax diagrams.

ENDCICS

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The DELAY parameter is only valid when OPTION(*CNTRLD) is specified.
- 2 All parameters preceding this point can be specified positionally.

Function

The ENDCICS CL command is used to initiate a request to end a specified control region and control the way in which work being processed by that control region and its associated shells is shut down.

In its simplest form, this command consists of:

```
ENDCICS CTLRGN(sysid)
```

Once a request to end a control region has been initiated, no new shells may be started for that control region until shutdown has completed and the control region has been restarted.

During control region shutdown, CICS/400 initiates requests to end all the associated shells (user or batch). In a "controlled" shutdown, specifying OPTION(*CNTRLD), the control region waits for the time specified on the DELAY option (by default the wait is indefinite) for acknowledgments from all shells that shutdown processing is complete. When the wait time expires or if an "immediate" shutdown, specifying OPTION(*IMMED), is selected, the control region completes shutdown processing and any shells remaining active may abend or report unexpected return codes.

For further information on shell shutdown, including information on ending a shell without shutting down the control region, refer to “Shell shutdown” on page 124 and “Ending a user shell (ENDCICSUSR)” on page 127.

Required parameters

CTLRGN

The name of the CICS control region to be shut down, also known as the CICS system ID.

control-region: The name of the control region may be up to four characters in length. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

OPTION

Identifies the method used to shut down the CICS control region.

Possible values are:

***CNTRLD:** All associated CICS shells are automatically shut down if they are idle or when the CICS transaction that is currently being executed is completed.

***IMMED:** All associated CICS shells are shut down immediately. Any CICS transactions that are being executed are not allowed to perform any cleanup. This parameter might cause loss of data; therefore, it should be used only after a controlled shutdown has been unsuccessful.

If this fails to shut down the control region, use the ENDJOB command. It may be that a shell has locked a part of the control region, in which case the shell will also need to be closed using an ENDJOB command. If, after 10 minutes, either the shell or the control region has not ended, issue an ENDJOBABN command for both the control region and the shell.

DELAY

Indicates the amount of time (in seconds) that is allowed to complete the controlled CICS control region shutdown. At the end of this time, or after 24 hours, whichever is the shorter, the control region is shut down immediately, and all shells associated with the control region are ended.

Possible values are:

***NOLIMIT:** The amount of time in which to complete a controlled shutdown is not limited. If the controlled shutdown has not completed after 24 hours, an immediate shutdown will take place.

time-delay: Valid values are in the range 1 through 99 999.

Examples

The ENDCICS command issues a controlled shutdown of the CICS control region called “TEST”. The controlled shutdown lasts for 360 seconds. If in that time all associated CICS shells have not shut down, then an immediate shutdown will be issued automatically.

```
ENDCICS CTLRGN(TEST) OPTION(*CNTRLD) DELAY(360)
```

ENDCICS

Chapter 9. Administering a CICS/400 shell

This chapter describes the following topics:

- “Introduction to the CICS/400 shell”
 - “Shell objects”
- “Shell processing” on page 122
 - “Shell initialization” on page 122
 - “Runtime processing” on page 123
 - “Shell shutdown” on page 124
- “Starting a user shell (STRCICSUSR)” on page 125
- “Ending a user shell (ENDCICSUSR)” on page 127
- “CESF—CICS sign-off transaction” on page 129

For most users, a CICS/400 shell is run interactively within an interactive OS/400 job. Shells use data queues to communicate with the control region.

Introduction to the CICS/400 shell

The CICS/400 shell provides the task scheduling and work management mechanisms to build and refresh the application programming environment required for CICS transactions. The CICS/400 user shell provides CICS application programs with an interface to CICS-managed resources.

There are three types of shell that support CICS transaction processing:

- User shells, which are started when a user enters a STRCICSUSR command. User shells may be started in either interactive or batch jobs.
- Interval control batch shells, which are controlled by the interval control facilities of the control region. See “Interval control batch shells” on page 95.
- Inbound intersystem communication shells, which are started either by a “Start Prestart Job” (STRPJ) command issued at control region startup or by the in-bound OS/400 EVOKE processing.

Transactions are scheduled from four sources:

- Terminal input (including STRCICSUSR CL command)
- Automatic task initiation (ATI) (by transient data queue trigger levels)
- Interval control (IC) starts
- Inbound intersystem communication (ISC) requests

CICS terminal users either explicitly or implicitly (by user profile or OS/400 menu selection) cause a STRCICSUSR command to be entered. Automatic Task Initiation and interval control start requests are processed in a similar manner. Both are essentially timer-driven transactions.

Shell objects

As in the control region, each shell creates a number of OS/400 objects during shell initialization. These objects give the shell a mechanism for information exchange with its associated control region. Unlike the control region, all the objects created at shell initialization are created in the QTEMP library with architected CICS/400 names beginning with the “AEG” prefix.

The user shell objects are:

- An OS/400 data queue QTEMP/AEGCICS *DATQ used for data queue information exchange between the shell and the control region.
- Three OS/400 user space (*USRSPC) objects. The space objects contain the directory of CICS internal control blocks needed within the shell, CICS internal control blocks required by the shells, and a nonshared or user storage area used for shell nonshared storage requests.
- An OS/400 user space called AEGCICS. This object contains internal CICS/400 areas that give access to shared control region information for all shells connected to the control region.

The QTEMP/AEGCICS *DTAQ shell object, QGPL/AEGxxxx user space, and the control region QTEMP/AEGCICS *DTAQ together create the link necessary for the exchange of information between the control region and its associated shells.

Shell processing

This section contains a more detailed description of the user shell processes. Besides providing an overall understanding, this section could be useful in tracking down an error condition.

There are three major user shell control processes:

- Shell initialization
- Shell runtime processing
- Shell shutdown

An unexpected return code during any of these processes causes message AEG1112 to be sent to the system operator, a dump to be taken, and a fatal error condition to be set.

Shell initialization

User application initialization builds the background environment common to all CICS sessions. This includes:

- Verifying that the CICS control region is up and available
- Invoking shell resource initialization routines
- Establishing a protected environment to insulate CICS from application errors
- Optionally, starting an initial transaction

In more detail, the initialization process consists of the following steps:

1. CICS verifies that this shell's user space does not exist. (If it exists, message AEG1114 is issued.)
2. The shell task control area is initialized.
3. The control region's user space is located, and some system information from the control region's area is copied into the shell's user space area.

Note: The shell's user space acts as an anchor control block that allows the EXEC interface program (called by application programs) to reestablish addressability (by using pointers) to internal control blocks.

You should note that by using the ENDSBS command rather than ENDCICS command, CICS is no longer in control of ending shell jobs. If there are active CICS shells when ENDSBS is issued, they may lose

contact with the control region because the control region has ended or the control region may try to contact a shell job which no longer exists. The result is the extraneous message may be issued in this case due to this lost communication.

4. An initialization and termination table, similar to the one used in the control region, is processed. For example, storage services are called first to set up user space areas. The return code passed back from each function is checked before proceeding with the initialization process. If errors occurred during shell initialization, message AEG1104 is issued (Initialization module_function _failed with return code _). A severity level of 20 or higher (message AEG1103 issued) causes shell initialization to cease.
5. A data queue request is sent to change the shell's dispatch control area (DCA) to a ready status.
6. If initial data, or an initial TRANID, or both, were specified in STRCICSUSR, the terminal control facility is called to build the initial terminal input/output area (TIOA).
7. If there was no initial TRANID, the message "CICS is ready for transactions" is sent to the terminal.

When all shell startup processes are complete, the terminal operator (if this is an interactive job) is sent a message indicating that the shell is initialized and operational. The CICS user can then enter transactions.

Runtime processing

After initialization, the shell coordinates the repetitive process of event scheduling, (ATI, IC START, ISC, or terminal input) scheduling and task termination with syncpoint and rollback processing.

Note: The shell can receive work from a **data queue, display file, or ICF file**.

The runtime processing of the shell can be described in the following steps:

1. The shell checks for timer event controlled activity.
2. The shell calls terminal control to accept input. This may be display file input, an intercommunication function, or a service request from the control region.
3. If a CICS data queue message is returned, the internal CICS function is called.
4. If a terminal input/output area (TIOA) is returned, CICS:
 - Determines a transaction id.
 - Ensures that the transaction is valid (that is, matches a PCT entry) and available.
 - Initializes the EIB.
 - Links to the application program.
 - If this is the first call, addressability to control blocks is established.
 - If a transaction work area (TWA) was specified, this area is allocated from nonshared GETMAIN storage.
 - If no syncpoint was requested by the program, the contents of the EIBRESP field of the EXEC interface block are tested. If EIBRESP is blank, the task has ended normally and CICS will perform a default syncpoint commit. If EIBRESP contains a return code, an abend has occurred and CICS will perform a rollback. See the *CICS for iSeries Application Programming Guide* for more information about EIBRESP.

If an application transaction abends, message AEG1111 is sent to the user indicating that the transaction abended.

- Performs a scan of task termination deferred work elements (DWEs) for work ready to process.
5. The shell checks whether CICS is in shutdown mode. If not, processing is repeated until shutdown is indicated.

Shell shutdown

Shutdown processing can be initiated by any of the following:

- Terminal input (CESF transaction).
- A single-shot transaction being specified. That is, an initial transaction was entered for the TRANID parameter on the STRCICSUSR command.
- Completion of the ATI or IC START work (nonterminal shells).
- ISC link shutdown.
- CICS control region shutdown requests.
- User shell shutdown requests (ENDJOB, ENDSBS, or ENDSYS CL commands).

Note: A shell will respond to an external end request, such as a message sent by the control region during shutdown processing, only when it next returns to a dequeue state. If the shell is processing a conversational task or is running in a held OS/400 job, shutdown will not occur immediately.

Shutdown provides for a reversal of the initialization process. A controlled shell shutdown consists of the following steps:

1. The terminal user (if any) is notified that shell shutdown has commenced.
2. A data queue message is sent to the control region to change the status to indicate that shutdown has begun.
3. The shell's control region user space is deleted.
4. In a similar manner to initialization, the termination modules list is processed to release resources and clean up the environment. The return code is checked after each module. If a severity of 20 or higher is returned (message AEG1109), the shutdown process is abandoned.
5. The control region is notified that the shell has completed the shutdown process.

If you want the terminal shell to shut down when either the terminal is switched off or the session is ended, the job description for the job running in the terminal shell should have the device recovery action (DEVRCYACN) parameter set to *ENDJOB. Some directly-connected terminals do not require this parameter to be set, but setting it does not change the shell execution. If the shell is to be run in a passthru session, the DEVRCYACN parameter must be set in the job description for the job that issues the STRPASTHR command.

If CESF LOGOFF (supplied transaction) is used, the shell logs the user off from CICS and issues the OS/400 SIGNOFF CL command.

If termination is caused by a CICS shutdown, the user is informed of the involuntary shutdown. Control is returned to OS/400 facilities.

Nonterminal users are shut down by internal data queue commands or ISC conversation link termination.

Starting a user shell (STRCICSUSR)

The STRCICSUSR CL command starts a CICS user shell associated with an active CICS control region for the OS/400 user who initiates this command. This command can be used by anyone on the OS/400 to start CICS transactions.

Some users may implicitly start this command by choosing a menu option that causes this command to be executed.

In its simplest form, this command consists of:

```
STRCICSUSR CTLRGN(sysid)
```

This starts a shell that is associated with a specified control region, and therefore runs under this control region. Changing the sysid enables you to associate the shell with any available control region.

You should avoid invoking interactive CICS shells from within a COBOL program as this may cause unpredictable results during exceptions after the shell has terminated.

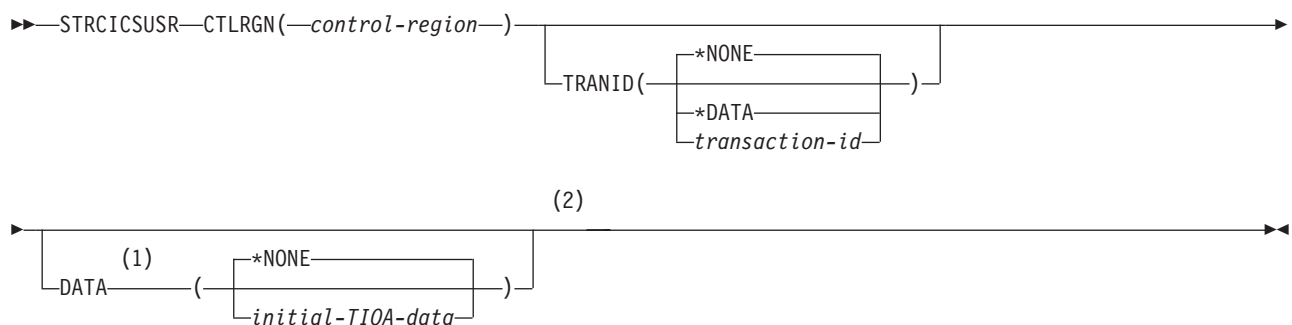
The full syntax diagram and the description of each parameter for this command follows. See “Interpreting the syntax diagrams” on page 133 for an explanation of the conventions of syntax diagrams.

CL command defaults

The defaults given in the CL command description are those that are supplied with the OS/400 system. You should check that your installation has not made any changes to these command default parameters.

STRCICSUSR

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The DATA parameter is required when TRANID(*DATA) is specified.
- 2 All parameters preceding this point can be specified positionally.

Function

The start CICS user (STRCICSUSR) command starts a CICS user shell associated with an active CICS control region for the OS/400 user who initiated this

STRCICSUSR

command. The command also specifies whether a CICS transaction is to be initiated when the CICS user shell has been started.

Required parameters

CTLRGN

The name of the CICS control region with which the CICS user shell will be associated. The name of the CICS control region is also known as the CICS system ID.

control-region: The name of the control region may be up to four characters in length. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

TRANID

The name of the CICS transaction identifier used to initiate a CICS program identifier defined in the CICS processing program table.

Possible values are:

***NONE:** The default CICS screen (which is blank) is displayed, prompting the CICS user to enter the CICS four-character transaction identifier.

When the CICS transaction is completed, the CICS user shell will stay active, waiting for the next transaction from the user. This process is known as a *multishot* CICS transaction environment.

***DATA:** Specifies that the first four characters of the DATA parameter are used as the CICS transaction identifier, to be the first transaction. This too is a multishot CICS transaction environment, because the shell remains active after the initial transaction is completed.

transaction-id: The transaction identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters \$, @, or #. The remaining characters can be alphanumeric or one of the special characters \$, @, or #. Lowercase transaction identifiers entered in this parameter are converted to uppercase; it is not possible to enter lowercase transaction identifiers.

Use of this value means that when this CICS transaction is completed, the CICS user shell returns to the point at which the STRCICSUSR command was entered, and no other transaction is executed. This process is known as a *single-shot* CICS transaction environment.

DATA

Indicates the data that is to be used to prime the CICS terminal input/output area (TIOA) for the first RECEIVE command.

Possible values are:

***NONE:** No data is passed to the CICS transaction.

initial-TIOA-data: Can be up to 3000 characters. The following restrictions apply to this value:

- If the first characters equal *CICS, the results will be unpredictable.
- If TRANID(*DATA) is specified, the first four characters will be taken as the transaction identifier.

Examples

The command

```
STRCICSUSR CTLRGN(PURC) TRANID(ORDR) DATA(ORDR 678BROWN99)
```

starts a shell in control region PURC and executes transaction ORDR in a single-shot transaction environment, passing "ORDR 678BROWN99" as the initial TIOA data. The TIOA data, in this example, includes the transaction ID so as to be identical in structure to the TIOA data that would typically result from interactive initiation of the transaction.

Ending a user shell (ENDCICSUSR)

The ENDCICSUSR CL command shuts down the specified shell and specifies what is to happen to active work being processed by that shell. Only the specified CICS shell is ended: the control region remains active, as well as other user shells.

You should use the ENDCICSUSR command if you are on the OS/400, but not logged onto the CICS shell you are trying to end. This command is most often used by a system administrator when there is a problem with a user shell. Otherwise, you should use the sign-off transaction CESF. See "CESF—CICS sign-off transaction" on page 129.

In its simplest form, this command consists of:

```
ENDCICSUSR JOB(job-name) CTLRGN(sysid)
```

The job name must be qualified enough to be unique (job name, user name, job number).

The CTLRGN(sysid) must refer to an active control region.

The full syntax diagram and the description of each parameter follow. See "Interpreting the syntax diagrams" on page 133 for an explanation of the conventions of syntax diagrams.

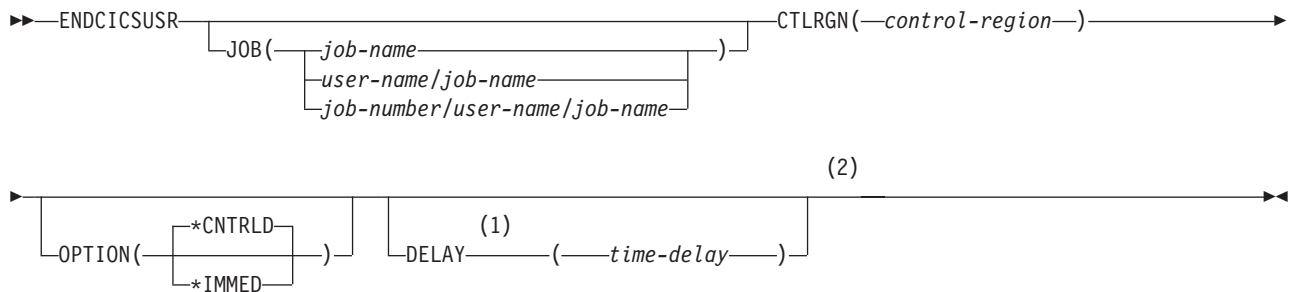
CL command defaults

The defaults given in the CL command description are those that are supplied with the OS/400 system. You should check that your installation has not made any changes to these command default parameters.

ENDCICSUSR

ENDCICSUSR

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The DELAY parameter is only valid when OPTION(*CNTRLD) is specified.
- 2 All parameters preceding this point can be specified positionally.

Function

The end CICS user (ENDCICSUSR) command shuts down a CICS shell and specifies what happens to active work being processed by that CICS shell.

Only the CICS shell is ended. The OS/400 job associated with the shell will still be active. For example, if an OS/400 user issued the STRCICSUSR command from the main menu, and then issued ENDCICSUSR from another OS/400 session, the OS/400 session that was used to issue the STRCICSUSR would be returned to the main menu, provided that the menu was not a CICS/400 transaction.

Required parameters

JOB

The name of the OS/400 job associated with the CICS shell that is to be shut down. If no CICS shell job qualifier is given, all of the OS/400 jobs currently in the OS/400 are searched for the simple OS/400 job name. If more than one of the specified OS/400 job names is found, a message is issued and a qualified OS/400 job name must be specified.

Possible values are:

job-name: The name of the OS/400 job associated with the CICS shell that is to be shut down.

user-name: The name of the OS/400 user associated with the CICS shell that is to be shut down.

job-number: The number of the OS/400 job that is associated with the CICS shell that is to be shut down.

CTLRGN

The name of the CICS control region associated with the CICS shell. The name of the control region is also known as the CICS system ID.

control-region: The length is four characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

OPTION

Identifies the method used to shut down the CICS shell.

Possible values are:

***CNTRLD:** The CICS shell will be shut down if it is idle or when the CICS transaction that is currently being executed is completed.

***IMMED:** The CICS shell is shut down immediately. Any CICS transaction that is being executed is not allowed to perform any cleanup. This option might cause undesirable results and, therefore, should be used only after a controlled shutdown has been unsuccessful.

DELAY

Indicates the amount of time (in seconds) that is allowed to complete the controlled CICS shell shutdown. If this amount of time is exceeded and the CICS shell shutdown is not complete, the CICS shell is shut down immediately.

time-delay: Valid values are in the range 1 through 99 999.

Examples

The ENDCICSUSR command issues a controlled shutdown of a CICS shell associated with the CICS control region PURC.

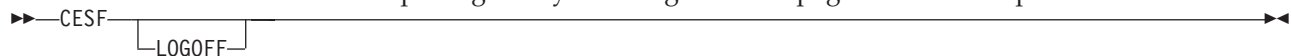
```
ENDCICSUSR JOB(001234/SMITH/XYZ) CTLRGN(PURC)
```

CESF—CICS sign-off transaction

The CICS CESF transaction is used to sign off from a user shell while still in CICS. It ends the user's own interactive shell.

The format of the CICS CESF transaction is:

See "Interpreting the syntax diagrams" on page 133 for an explanation of the



conventions of syntax diagrams.

CESF LOGOFF causes the user to be logged off from both CICS and the OS/400.

If the OS/400 STRCICSUSR CL command was used to sign on to the CICS user shell, the CICS user is returned at sign off to the OS/400 session. Otherwise, the CICS user is disconnected from the OS/400 system.

ENDCICSUSR

Part 3. CICS resources

Chapter 10. Defining resources-reference information.

Interpreting the syntax diagrams	133
Managing groups	135
Using the CRTICISGRP command	135
Using the CHGCISGRP command	137
Using the DLTCISGRP command	138
Using the INSCISGRP command	138
Using the SAVCISGRP command	140
Using the WRKCISGRP command	141
Managing data conversion resource definitions	142
Using the ADDCISCVT command	143
Using the CHGCISCVT command	149
Using the DSPCISCVT command	155
Using the RMVCISCVT command	157
Using the WRKCISCVT command	159
Managing destination definitions	160
Using the ADDCISDCT command	161
Using the CHGCISDCT command	167
Using the DSPCISDCT command	173
Using the RMVCISDCT command	175
Using the WRKCISDCT command	176
Managing file resource definitions	177
Using the ADDCISFCT command	177
Using the CHGCISFCT command	183
Using the DSPCISFCT command	189
Using the RMVCISFCT command	191
Using the WRKCISFCT command	192
Managing group list resource definitions	193
Using the ADDCISGLT command	194
Using the DSPCISGLT command	196
Using the RMVCISGLT command	198
Using the WRKCISGLT command	200
Managing journal resource definitions	201
Using the ADDCISJCT command	202
Using the CHGCISJCT command	206
Using the DSPCISJCT command	209
Using the RMVCISJCT command	210
Using the WRKCISJCT command	212
Managing transaction definitions	213
Using the ADDCISPCT command	214
Using the CHGCISPCT command	218
Using the DSPCISPCT command	222
Using the RMVCISPCT command	223
Using the WRKCISPCT command	225
Managing program definitions	226
Using the ADDCISPPT command	227
Using the CHGCISPPT command	231
Using the DSPCISPPT command	235
Using the RMVCISPPT command	236
Using the WRKCISPPT command	238
Managing system initialization resource definitions	239
Using the ADDCISST command	239
Using the CHGCISST command	248
Using the DSPCISST command	258
Using the RMVCISST command	259

Using the WRKCISST command	260
Managing system resource definitions	260
Using the ADDCICSTCS command	261
Using the CHGCICSTCS command	265
Using the DSPCICSTCS command	269
Using the RMVCICSTCS command	270
Using the WRKCICSTCS command	272
Managing terminal resource definitions	273
Uppercase conversion of terminal input	273
Datastream compression	274
Using the ADDCICSTCT command	275
Using the CHGCICSTCT command	282
Using the DSPCICSTCT command	291
Using the RMVCICSTCT command	292
Using the WRKCICSTCT command	294
Managing temporary storage resource definitions	295
Using the ADDCICSTST command	296
Using the CHGCICSTST command	298
Using the DSPCICSTST command	300
Using the RMVCICSTST command	301
Using the WRKCICSTST command	302

Chapter 11. Autoinstall for terminal definitions 305

How autoinstall terminal identifiers are generated	307
Defining an autoinstall model terminal using masking	308
Using an autoinstall control program	309
The CICS/400-supplied autoinstall control program	309
How does CICS/400 know to call an autoinstall control program?	309
Creating a PPT definition for AEGTCACP	309
Translating the autoinstall control program	310
Security	311
Program object security	311
Autoinstall delete security considerations	311
Batch shell considerations	311
Storage considerations	311
CEMT INQUIRE/SET PROGRAM and EXEC CICS INQUIRE/SET PROGRAM	312
Turning off autoinstall in an active CICS system	312
Attempting to install an AEGTCACP PPT entry defined as remote	312
Writing your own autoinstall program	312
Autoinstalling terminal definitions	313
The COMMAREA for an install request	313
Returning information to CICS/400	315
Selecting the autoinstall model	316
Setting the terminal name	316
CICS/400 action on return from the control program	317
Deleting autoinstalled terminal definitions	317
The COMMAREA for a delete request	318
Naming your control program	318
Testing and debugging your control program	318
Customizing the sample program	319

Chapter 10. Defining resources-reference information

This chapter describes each of the control language (CL) commands that are used to define CICS resources. Groups are dealt with first, followed by the resource definition tables in alphabetical order.

This chapter covers the following topics:

- “Interpreting the syntax diagrams”
- “Managing groups” on page 135
- “Managing data conversion resource definitions” on page 142
- “Managing destination definitions” on page 160
- “Managing file resource definitions” on page 177
- “Managing group list resource definitions” on page 193
- “Managing journal resource definitions” on page 201
- “Managing transaction definitions” on page 213
- “Managing program definitions” on page 226
- “Managing system initialization resource definitions” on page 239
- “Managing system resource definitions” on page 260
- “Managing terminal resource definitions” on page 273
- “Managing temporary storage resource definitions” on page 295

Guidance information on resource definition can be found in Chapter 4, “Defining resources” on page 41.

Interpreting the syntax diagrams

Table 23 explains the command syntax conventions. You interpret the syntax by following the arrows from left to right.

Table 23. Command syntax conventions

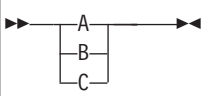
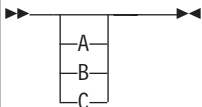
Symbol	Action
	A set of alternatives—one of which you <i>must</i> code.
	A set of alternatives—one of which you <i>may</i> code.

Table 23. Command syntax conventions (continued)

Symbol	Action
	A set of alternatives—any of which you may code. If applicable, the syntax shows the number of times the parameter may be repeated.
	Alternatives where A is the default.
<p>Name:</p>	Use with the named section in place of its name.
Uppercase characters and brackets	Code exactly as shown. You may use either uppercase or lowercase letters.
Lowercase characters appearing like <i>this</i>	Code your own text, as appropriate (for example, <i>name</i>).

For example, with **GROUP**(*group-name*), you must code **GROUP** and **()** unchanged, but are free to code any valid text string to mean the name of the group.

Some parameters for a command may be entered without their keywords. The system determines which parameter is indicated by its position in the list. Again, refer to the relevant syntax diagrams for details.

For parameters that have more than one element, you must enter all elements, including defaults, leaving a blank space between elements. All parameters are shown in each diagram in the order required by the system for positional coding.

All required parameters precede all optional parameters. The required parameters (if any) are boxed with the command name at the beginning of the diagram. All the other parameters are optional and do not have to be coded; a default value (shown above the line) is assumed for each uncoded parameter for most commands.

For each parameter that can have a repetition of values, the maximum number of repetitions that can be coded is shown in the diagram with the parameter's values. The syntax diagrams also show (by the use of flow lines and notes) which parameters are dependent on the values of other parameters (mutually dependent) and which can be used only if another parameter or value does not cause a conflict (mutually exclusive).

Entry codes shown in the upper right corner of the diagram indicate the environment in which the command can be specified. Notes are also included to give information needed to properly interpret the syntax.

For more detailed information on syntax diagrams, refer to the CL topic in the iSeries Information Center.

CL command defaults

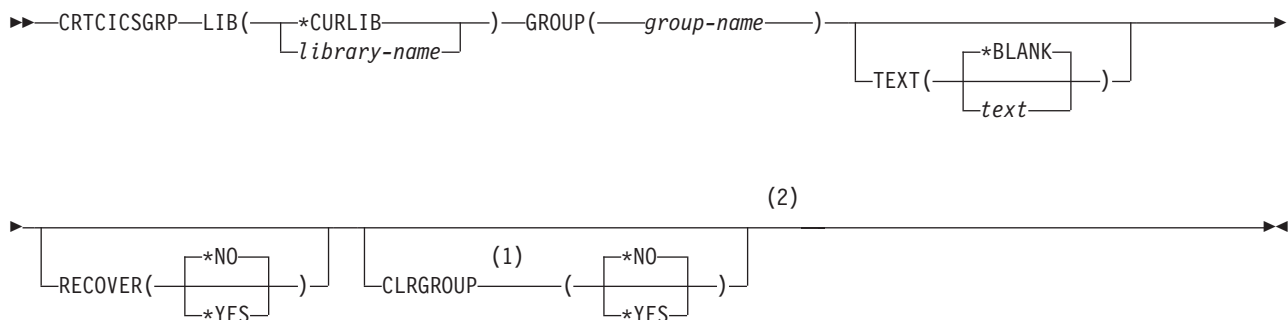
The defaults given in the CL command descriptions are those that are supplied with the OS/400 system. You should check that your installation has not made any changes to these command default parameters.

Managing groups

A resource definition group comprises a number of resource definitions that are used together in a CICS control region. You should create a group before you create the resource definitions for that group. The group name should go in the system initialization table.

Using the CRTICSGRP command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 This parameter is valid only when RECOVER(*YES) is specified.
- 2 All parameters preceding this point can be specified positionally.

Function

Use the Create CICS/400 Group (CRTICSGRP) command to create all tables associated with a group.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that will contain the group.

Possible values are:

***CURLIB:** The current library will contain the group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that will contain the group.

Group (GROUP)

Enter the name of the group to be created.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

Description (TEXT)

Enter the text that describes the group.

Possible values are:

***BLANK:** No text will be associated with the group.

text: Specify a character string of up to 50 characters to describe the group. Enclose the string in apostrophes to use leading or trailing blanks.

Recover (RECOVER)

Specifies whether or not to attempt to restore the group that has been previously damaged.

Note: All this function does is make sure that all tables are available to this group. Therefore, this parameter only has an effect on an already-existing group.

Possible values are:

***NO:** Do not attempt to recover the group.

***YES:** Attempt to recover the group.

Clear file (CLRGROUP)

Specifies whether or not to erase all entries from all tables for this group.

Note: This parameter only has an effect on an already-existing group.

Possible values are:

***NO:** Do not clear the group.

***YES:** Clear the group.

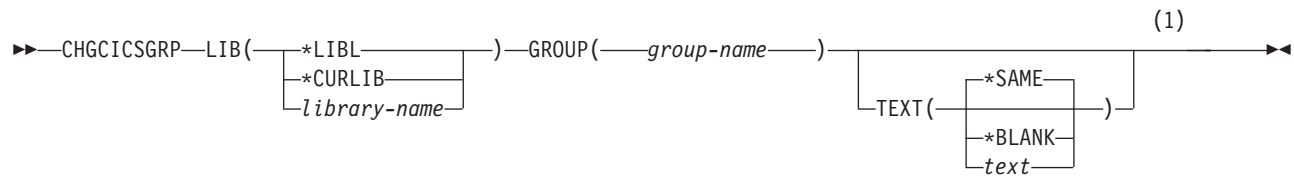
Examples

```
CRTCICSGRP LIB(CICSWORK) GROUP(ACCT) TEXT('CICS/400
sample applications group.')
```

This command creates a group ACCT located in OS/400 library CICSWORK that will hold the resource definitions for the sample transaction ACCT.

Using the CHGCICSGRP command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Change CICS/400 Group (CHGCICSGRP) command to change the text describing all tables associated with a group.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to be changed.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

Description (TEXT)

Enter the text that describes the group.

Possible values are:

***SAME:** The text associated with the group is not changed.

***BLANK:** No text will be associated with the group.

text: Specify a character string of up to 50 characters to describe the group. Enclose the string in apostrophes to use leading or trailing blanks.

Examples

```
CHGCICSGRP LIB(CICSWORK) GROUP(ACCT) TEXT('CICS/400
sample applications group.')
```

This command changes the descriptive text of resource definitions created for group ACCT in OS/400 library CICSWORK.

DLTCICSGRP

Using the DLTCICSGRP command

Job: B,I Pgm: B,I REXX: B,I Exec

```
DLTCICSGRP LIB( *LIBL ) GROUP( group-name ) (1)
```

The diagram shows the command syntax with a bracket indicating that the parameters `*LIBL`, `*CURLIB`, and `library-name` are grouped together as the argument for the `LIB` parameter.

Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Delete CICS/400 Group (DLTCICSGRP) command to delete all tables associated with a group. The tables may be at any release level.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to be deleted.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Examples

```
DLTCICSGRP LIB(CICSWORK) GROUP(ACCT)
```

This command deletes group ACCT located in OS/400 library CICSWORK.

Using the INSCICSGRP command

Job: B,I Pgm: B,I REXX: B,I Exec

```
INSCICSGRP CTRLGN( control-region ) LIB( *LIBL ) (1)
```

The diagram shows the command syntax with a bracket indicating that the parameters `*LIBL`, `*CURLIB`, and `library-name` are grouped together as the argument for the `LIB` parameter.

```
GROUP( group-name ) (1)
```

Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Install CICS/400 Group (INSCICSGRP) command to define all entries in the tables within the group, to the runtime resource table definitions for the active CICS/400 control region. This command is run as a batch job to install dynamically both new and amended resource definition entries into the CVT, PCT, PPT, TCS, and TCT. Up to 200 entries of each type may be installed using this command.

If the table entry is already defined in the runtime resource table definitions, via either the control region start (STRCICS command) or a previous INSCICSGRP command issued for the control region, the table entry replaces the existing runtime resource table definition if it is currently not in use.

You can install new entries, but not amended entries, for the following tables:

- Destination control table
- File control table
- Journal control table

The INSCICSGRP CL command ignores the following tables in the group:

- Group list table
- System initialization table
- Temporary storage table

Required parameters

CICS control region (CTLRGN)

Enter the name of the active CICS/400 control region that the table entries are to be defined to the runtime resource table definitions.

control-region: The control region name may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL**: The library list is used to locate the first OS/400 library that contains the group.

***CURLIB**: The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to be installed.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

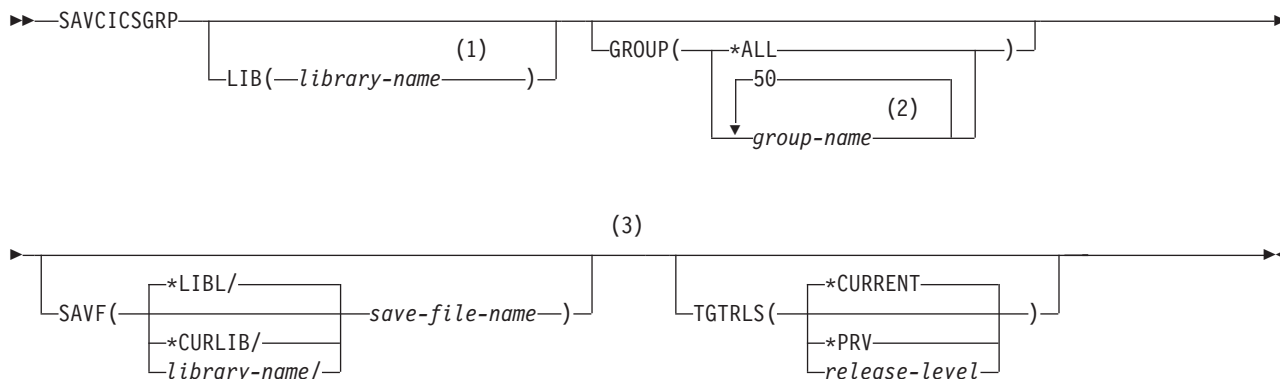
Examples

```
INSCICSGRP CTLRGN(PURC) LIB(CICSWORK)
GROUP(ACCT)
```

This command defines the table entries located in group ACCT in OS/400 library CICSWORK to the runtime resource definition for the active CICS/400 control region PURC.

Using the SAVCICSGRP command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 LIB(QTEMP) may not be used with TGTRLS(*PRV).
- 2 A maximum of 50 repetitions.
- 3 All parameters preceding this point can be specified positionally.

Function

The Save CICS/400 Groups (SAVCICSGRP) command causes the specified groups in the library to be saved by using the SAVOBJ system command. The objects can be saved for use on either the current release or a supported previous release. The supported previous releases are the last release of the previous version and the previous release of the current version. The required release is specified in the TGTRLS parameter.

When TGTRLS has a previous release value, resource definition tables for the specified groups are converted to the previous release of CICS/400 and placed in the QTEMP library. The resource definitions in the QTEMP library are saved in the file specified in the SAVF parameter.

Groups containing resource tables that specify features that are not supported in the previous release are not saved.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the tables in a specific group(s) to be saved.

library-name: Specify the name of the OS/400 library that contains the groups.

Group (GROUP)

Enter the name of the group(s) to be saved.

Possible values are:

***ALL:** All groups in the library will be saved (except those containing resource tables that specify features that are not supported on the target release).

group-name: The name of the group may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or one of the special characters, \$, @, or #.

Save file (SAVF)

Enter the name of the save file (in library-name/save-file-name format) that is used to contain the saved groups. The save file must be empty, that is, either newly allocated or cleared using the CLRSAVF CL command. Otherwise the SAVOBJ command will fail.

Possible values are:

***LIBL**: The library list is used to locate the save file.

***CURLIB**: The current library for the job is used to locate the save file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the save file is located.

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

Target release (TGTRLS)

Enter the release of CICS/400 on which you intend to save the object.

Possible values are:

***CURRENT**: The object is to be used on the release of CICS/400 currently running on your system. For example, if Version 5 Release 2 is running on the system, *CURRENT means you intend to use the object on a system with Version 5 Release 2 installed.

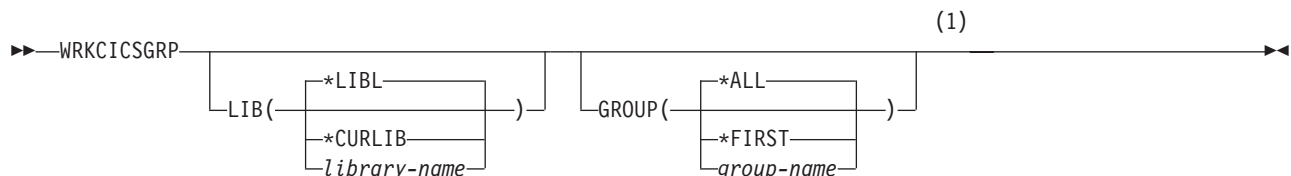
***PRV**: The object is to be used on a supported previous release of the CICS/400 system. For example, if Version 5 Release 2 is running on your system, *PRV means you intend to use the object on a system with Version 5 Release 1 installed.

release-level: Specify the release in the format VxRxMx, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V5R1M0 is version 5, release 1, modification level 0. The object can be used on a system with the specified release.

Valid values depend on the current version, release, and modification level, and they change with each new release.

Using the WRKCICSGRP command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Group (WRKCICSGRP) command to list all groups in the OS/400 library. This command allows new groups to be created, and existing groups to be changed, deleted, installed, recovered, and saved.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate all of the groups that are in the OS/400 libraries specified in the OS/400 library list.

***CURLIB:** The current library contains the group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to work with.

Possible values are:

***ALL:** All the groups that are associated with the OS/400 library are listed.

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Examples

```
WRKCICSGRP LIB(CICSWORK)
```

This command lists group ACCT in library CICSWORK.

Managing data conversion resource definitions

Conversion vector table (CVT) entries define templates for data conversion either from EBCDIC to ASCII, or when the code page and character set used by the other CICS system is different from that used by the OS/400 system. The data conversion must be defined by the system sending the data. Data conversion is required for:

- File control commands
- Transient data queue commands
- Temporary storage queue commands
- START commands
- LINK commands

Data is converted according to templates that have been predefined in the CVT. A template describes the type of data, how it is arranged in the record, and the type of conversion required. If your data contains alphabetic and numeric fields, you may need to define a template for each field in the record.

There are three types of conversion template:

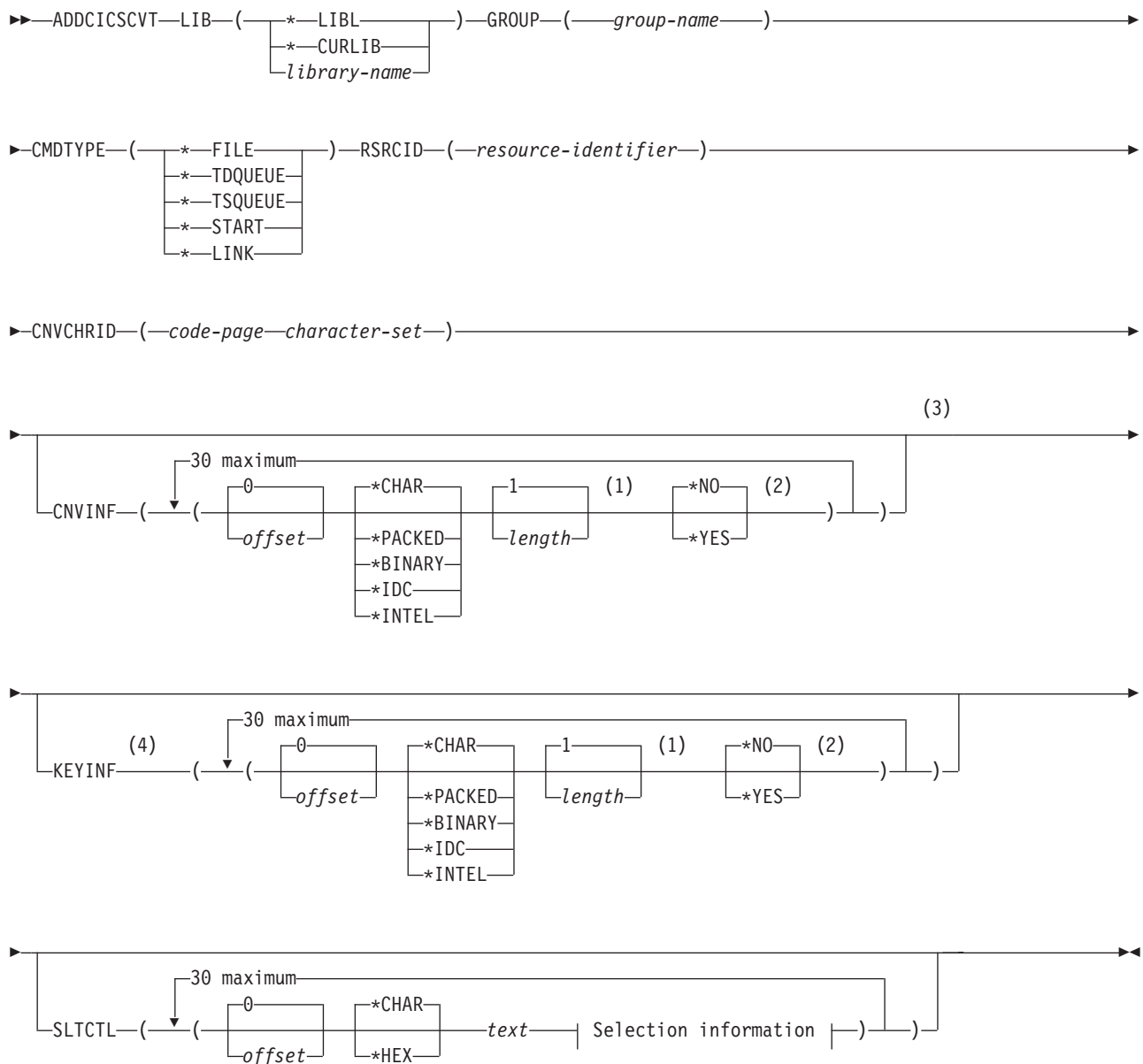
- For conversion of file keys, use the **Key conversion data (KEYINF)** parameter. This type of conversion is valid only for key-sequenced files.

- For conversion of variable format data in accordance with defined selection criteria, use the **Selection criteria (SLTCTL)** parameter.
- For default conversion to be applied when no other conversion template matches the data, use the **Conversion information (CNVINF)** parameter.

See *CICS for iSeries Intercommunication* for information about intersystem communication.

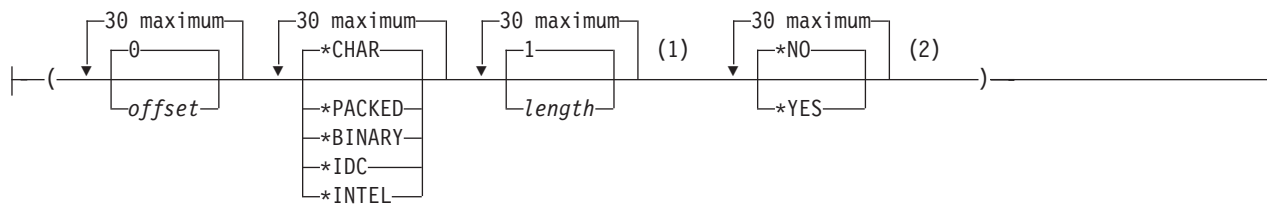
Using the ADDCICSCVT command

Job: B,I Pgm: B,I REXX: B,I Exec



ADDCICSCVT

Selection information:



Notes:

- 1 The Element 3 (Length) value must be 2 or 4 when Element 2 (Type) is *INTEL.
- 2 Element 4 (SO/SI) is valid only when Element 2 (Type) is *CHAR.
- 3 All parameters preceding this point can be specified positionally.
- 4 The KEYINF parameter is valid only when CMDTYPE(*FILE) is specified.

Function

Use the Add CICS/400 Conversion Vector Table (ADDCICSCVT) command to add an entry in the CVT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this CVT entry is to be added.

group-name: The group name may have a maximum length of 10 characters. In all cases, the first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

API command type (CMDTYPE)

Indicates the type of EXEC CICS command that will use this entry for user data conversion. This parameter is used with the **Resource identifier (RSRCID)** parameter to identify this CVT entry.

Possible values are:

- ***FILE:** File control commands.
- ***TDQUEUE:** Transient data queue commands.
- ***TSQUEUE:** Temporary storage queue commands.
- ***START:** The START command.
- ***LINK:** The LINK command.

Resource identifier (RSRCID)

Enter the resource identifier associated with the EXEC CICS command type. This parameter is used with the **Command type (CMDTYPE)** parameter to identify this CVT entry.

This resource identifier should have an entry in the appropriate table, as follows:

CMDTYPE	CICS/400 table
*TDQUEUE	Destination Control Table
*START	Program Control Table
*FILE	File Control Table
*TSQUEUE	Temporary Storage Table
*LINK	Processing Program Table

resource-identifier: For a command type of *TDQUEUE or *START, the maximum length of the resource identifier is 4 characters. For a command type of *FILE, *TSQUEUE, or *LINK, the maximum length of the resource identifier is 8 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

Character identifier (CNVCHRID)

Enter the code page and character set to be used to convert user data that is to be sent from or to another CICS system.

There are 2 elements to this field. Possible values are:

Element 1: Code page

The code page to be used by the conversion.

code-page: Enter a number in the range 1 through 65 535.

Element 2: Graphic character set

Enter the graphic character set to be used by the conversion.

character-set: Enter a number in the range 1 through 65 535.

Optional parameters

Conversion information (CNVINP)

This field defines a default conversion template. Use this field to define conversion information for data that does not satisfy the criteria specified in the **Selection criteria (SLTCLT)** parameter.

Note: Use the key information field for converting key fields.

Specify:

- The position of the data in the record
- The type of data in the field
- The length of the data
- Whether or not the data contains shift-out/shift-in (SO/SI) characters

There are 4 elements to this field. Possible values are:

Element 1: Starting location (offset)

The position in the record of the start of the data to be converted.

0: Conversion should start at the beginning of the record.

offset: A number in the range 0 through 65 535.

Element 2: Type of conversion

Type of conversion to be done on the field.

***CHAR**: The data is alphabetic and will be converted from the code page specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***PACKED**: The data is in packed decimal format and will not be converted.

***BINARY**: The data is in binary format and will not be converted.

***IDC**: The data is in ideographic format and will be converted from the character set specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***INTEL**: The data is in Intel** format and will be converted by reversing the bytes. The field can have a length of two or four only.

Element 3: Length of conversion

The length in bytes of the field to be converted.

1: A data length of one byte will be converted.

length: Enter a number in the range 1 through 65 535. If the data is in Intel format, this field must contain 2 or 4.

Element 4: User-specified DBCS data

This element, which is valid only when *CHAR is specified for the field, indicates whether or not the data to be converted contains DBCS characters. DBCS data should be indicated by SO/SI delimiters. The DBCS data will not be converted.

***NO**: The data does not have SO/SI characters.

***YES**: The data has SO/SI characters.

Key conversion data (KEYINF)

This field defines the conversion template for one key field. Specify:

- The position of the key field in the record
- The length of the key
- The type of conversion to be applied to the key

This parameter is valid only for a command type of *FILE, and if the file is accessed using keys; that is it is a key-sequenced data set (KSDS).

Note: The FCT entry defines the location of the key.

Possible values are:

Element 1: Starting location (offset)

The position in the record of the key field.

0: The key is at the beginning of the record.

offset: A number in the range 0 through 65 535 that gives the key position in bytes from the start of the record.

Element 2: Type of conversion

The type of conversion to be done on the field.

***CHAR**: The data is alphabetic and will be converted from the code page specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***PACKED:** The data is in packed decimal format and will not be converted.

***BINARY:** The data is in binary format and will not be converted.

***IDC:** The data is in ideographic format and will be converted from the character set specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***INTEL:** The data is in Intel format and will be converted by reversing the bytes. The field can have a length of two or four only.

Element 3: Length of conversion

The length in bytes of the field to be converted.

1: A data length of 1 byte will be converted.

length: Enter a number in the range 1 through 65 535. If the data is in Intel format, this field must contain 2 or 4.

Element 4: User-specified DBCS data

This element, which is valid only when *CHAR is specified for the field, indicates whether or not the data to be converted contains DBCS characters. DBCS data should be indicated by SO/SI delimiters. The DBCS data will not be converted.

***NO:** The data does not have SO/SI characters.

***YES:** The data has SO/SI characters.

Selection criteria (SLTCTL)

Enter the selection criteria and the conversion information for each field in the user data that is to be sent from or to another CICS system. Selection criteria consist of the position and value to be checked in the user data in order to use the associated conversion information. The conversion information consists of the position and length of the field and the type of conversion to be applied to it.

Up to 30 fields may be defined. However, unlike the CNVINP and KEYINF parameters, all occurrences of one element must be completed before entering the occurrences of the next element. You must enter the elements for the fields in the same order, so that CICS can make the necessary linkages for conversion. For example, if you are defining the selection criteria for five fields, you must enter all five offset positions, followed by all five formats, and so on, maintaining the order of the fields across the elements.

Note: If the user data does not satisfy the conditions that are specified by any of the selection criteria, then the **Conversion information (CNVINP)** field is used to convert the user data.

Possible values are:

Element 1: Starting location (offset)

The position in the record where the comparison of data should start.

0: The conversion should start at the beginning of the record.

offset: Enter a number in the range 0 through 65 535 that specifies the position in bytes of the start of the data.

Element 2: Character or hex format

Indicates whether the selection data is character or hexadecimal data.

***CHAR:** Character data.

***HEX:** Hexadecimal data.

ADDCICSCVT

Element 3: Value to compare against (text)

The value with which the data is to be compared, in order to use the associated conversion information.

text: Up to 254 alphanumeric or 127 hexadecimal characters.

Selection information

The conversion information for each field in the user data to be used when the selection criteria is a match. Possible values are:

Element 1: Starting location (offset)

The position in the record where conversion of data should start.

0: Conversion of data should start at the beginning of the record.

offset: A number in the range 0 through 65 535 that gives the position in bytes from the start of the record.

Element 2: Type of conversion

The type of conversion to be done on the field.

***CHAR**: The data is alphabetic and will be converted from the code page specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***PACKED**: The data is in packed decimal format and will not be converted.

***BINARY**: The data is in binary format and will not be converted.

***IDC**: The data is in ideographic format and will be converted from the character set specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***INTEL**: The data is in Intel format and will be converted by reversing the bytes. The field can have a length of two or four only.

Element 3: Length of conversion

The length in bytes of the field to be converted.

1: A data length of one byte is to be converted.

length: Enter a number in the range 1 through 65 535. If the data is in Intel format, this field must contain 2 or 4.

Element 4: User-specified DBCS data

This element, which is valid only when ***CHAR** is specified for the field, indicates whether or not the data to be converted contains DBCS characters. DBCS data should be indicated by SO/SI delimiters. The DBCS data will not be converted.

***NO**: The data does not have SO/SI characters.

***YES**: The data has SO/SI characters.

Examples

```
ADDCICSCVT LIB(PROD) GROUP(ACTPAY)
  CMDTYPE(*FILE)
  RSRCID(PAYABLES) CNVCHRID(1 1)
  CNVINFL((0 *CHAR 250 *NO))
```

This command adds a CVT entry called ***FILEPAYABLES** to group **ACTPAY** in OS/400 library **PROD**. The conversion of the user data that is sent from or to another CICS system will use the code page 1 and character set 1. It will convert user data starting in position 0, for a length of 250, as character data with no SO/SI delimiters.

Using the CHGCICSCVT command

Job: B,I Pgm: B,I REXX: B,I Exec

▶▶ CHGCICSCVT LIB ((* LIBL)) GROUP (*group-name*)

* CURLIB
library-name

▶ CMDTYPE ((* FILE)) RSRCID (*resource-identifier*)

* TDQUEUE
* TSQUEUE
* START
* LINK

▶ CNVCHRID ((* SAME) (* SAME))

code-page character-set

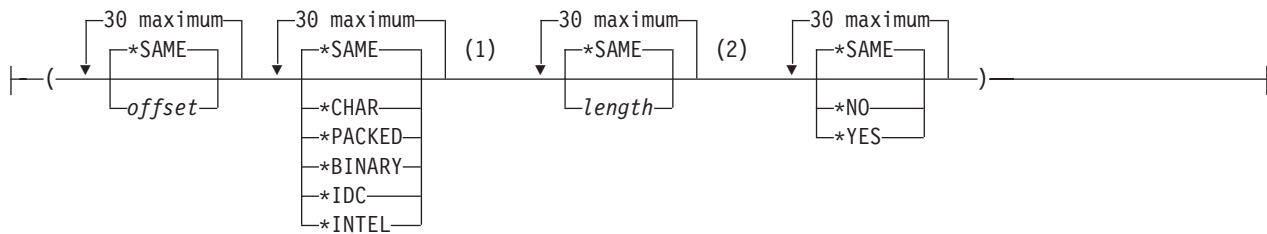
▶ CNVINFL ((* SAME (30 maximum) (* SAME (* CHAR * PACKED * BINARY * IDC * INTEL)) (* SAME (1) length) (* SAME (2) * NO * YES)))

▶ KEYINF (4) ((* SAME (30 maximum) (* SAME (* CHAR * PACKED * BINARY * IDC * INTEL)) (* SAME (1) length) (* SAME (2) * NO * YES)))

▶ SLTCTL ((* SAME (30 maximum) (* SAME (* CHAR * HEX)) (* SAME text) Selection information))

Selection information:

CHGCICSCVT



Notes:

- 1 The Element 3 (Length) value must be 2 or 4 when Element 2 (Type) is *INTEL.
- 2 Element 4 (SO/SI) is valid only when Element 2 (Type) is *CHAR.
- 3 All parameters preceding this point can be specified positionally.
- 4 The KEYINF parameter is valid only when CMDTYPE(*FILE) is specified.

Function

Use the Change CICS/400 Conversion Vector Table (CHGCICSCVT) command to change an entry in the CVT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the CVT entry to be changed.

group-name: The group name may have a maximum length of 10 characters. In all cases, the first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

API command type (CMDTYPE)

Indicates the type of EXEC CICS command that will use this entry for user data conversion. This parameter is used with the **Resource identifier (RSRCID)** parameter to identify this CVT entry.

Possible values are:

- ***FILE:** File control commands.
- ***TDQUEUE:** Transient data queue commands.
- ***TSQUEUE:** Temporary storage queue commands.
- ***START:** The START command.
- ***LINK:** The LINK command.

Resource identifier (RSRCID)

Enter the resource identifier associated with the EXEC CICS command type. This parameter is used with the **Command type (CMDTYPE)** parameter to identify this CVT entry.

This resource identifier should have an entry in the appropriate table, as follows:

CMDTYPE	CICS/400 table
*TDQUEUE	Destination Control Table
*START	Program Control Table
*FILE	File Control Table
*TSQUEUE	Temporary Storage Table
*LINK	Processing Program Table

resource-identifier: For a command type of *TDQUEUE or *START, the maximum length of the resource identifier is 4 characters. For a command type of *FILE, *TSQUEUE, or *LINK, the maximum length of the resource identifier is 8 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

Optional parameters

Character identifier (CNVCHRID)

Enter the code page and character set to be used to convert user data that is to be sent from or to another CICS system.

There are two elements to this field. Possible values are:

Element 1: Code page

The code page to be used by the conversion.

***SAME:** Keep the value currently specified in the CVT entry.

code-page: Enter a number in the range 1 through 65 535.

Element 2: Graphic character set

Enter the graphic character set to be used by the conversion.

***SAME:** Keep the value currently specified in the CVT entry.

character-set: Enter a number in the range 1 through 65 535.

Conversion information (CNVIN)

This field defines a default conversion template. Use this field to define conversion information for data that does not satisfy the criteria specified in the **Selection criteria (SLTCTL)** parameter.

Note: Use the key information field for converting key fields.

Specify:

- The position of the data in the record
- The type of data in the field
- The length of the data
- Whether or not the data contains shift out/shift in (SO/SI) characters

There are 4 elements to this field. Possible values are:

Element 1: Starting location (offset)

The position in the record of the start of the data to be converted.

***SAME:** Keep the value currently specified in the CVT entry.

offset: A number in the range 0 through 65 535.

Element 2: Type of conversion

Type of conversion to be done on the field.

***SAME:** Keep the value currently specified in the CVT entry.

***CHAR:** The data is alphabetic and will be converted from the code page specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***PACKED:** The data is in packed decimal format and will not be converted.

***BINARY:** The data is in binary format and will not be converted.

***IDC:** The data is in ideographic format and will be converted from the character set specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***INTEL:** The data is in Intel format and will be converted by reversing the bytes. The field can have a length of two or four only.

Element 3: Length of conversion

The length in bytes of the field to be converted.

***SAME:** Keep the value currently specified in the CVT entry.

length: Enter a number in the range 1 through 65 535. If the data is in Intel format, this field must contain 2 or 4.

Element 4: User-specified DBCS data

This element, which is valid only when *CHAR is specified for the field, indicates whether or not the data to be converted contains DBCS characters. DBCS data should be indicated by SO/SI delimiters. The DBCS data will not be converted.

***SAME:** The value currently specified in the CVT entry.

***NO:** The data does not have SO/SI characters.

***YES:** The data has SO/SI characters.

Key conversion data (KEYINF)

This field defines the conversion template for one key field. Specify:

- The position of the key field in the record
- The length of the key
- The type of conversion to be applied to the key

This parameter is valid only for a command type of *FILE, and if the file is accessed using keys; that is it is a key-sequenced data set (KSDS).

Note: The FCT entry defines the location of the key.

Possible values are:

Element 1: Starting location (offset)

The position in the record of the key field.

***SAME:** Keep the value currently specified in the CVT entry.

offset: A number in the range 0 through 65 535 that gives the key position in bytes from the start of the record.

Element 2: Type of conversion

The type of conversion to be done on the field.

***SAME:** Keep the value currently specified in the CVT entry.

***CHAR:** The data is alphabetic and will be converted from the code page specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***PACKED:** The data is in packed decimal format and will not be converted.

***BINARY:** The data is in binary format and will not be converted.

***IDC:** The data is in ideographic format and will be converted from the character set specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***INTEL:** The data is in Intel format and will be converted by reversing the bytes. The field can have a length of two or four only.

Element 3: Length of conversion

The length in bytes of the field to be converted.

***SAME:** Keep the value currently specified in the CVT entry.

length: Enter a number in the range 1 through 65 535. If the data is in Intel format, this field must contain 2 or 4.

Element 4: User-specified DBCS data

This element, which is valid only when *CHAR is specified for the field, indicates whether or not the data to be converted contains DBCS characters. DBCS data should be indicated by SO/SI delimiters. The DBCS data will not be converted.

***SAME:** Keep the value currently specified in the CVT entry.

***NO:** The data does not have SO/SI characters.

***YES:** The data has SO/SI characters.

Selection criteria (SLTCTL)

Enter the selection criteria and the conversion information for each field in the user data that is to be sent from or to another CICS system. Selection criteria consist of the position and value to be checked in the user data in order to use the associated conversion information. The conversion information consists of the position and length of the field and the type of conversion to be applied to it.

Note: If the user data does not satisfy the conditions that are specified by any of the selection criteria, then the **Conversion information (CNVINP)** field is used to convert the user data.

Possible values are:

Element 1: Starting location (offset)

The position in the record where the comparison or data should start.

***SAME:** Keep the value currently specified in the CVT entry.

offset: Enter a number in the range 0 through 65 535 that specifies the position in bytes of the start of the data.

Element 2: Character or hex format

Indicates whether the selection data is character or hexadecimal data.

***SAME:** Keep the value currently specified in the CVT entry.

***CHAR:** Character data.

***HEX:** Hexadecimal data.

Element 3: Value to compare against (text)

The value with which the data is to be compared, in order to use the associated conversion information.

***SAME:** Keep the value currently specified in the CVT entry.

text: Up to 254 alphanumeric characters or 127 hexadecimal characters.

Selection information

The conversion information for each field in the user data to be used when the selection criteria is a match. Possible values are:

Element 1: Starting location (offset)

The position in the record where conversion of data should start.

***SAME:** Keep the value currently specified in the CVT entry.

offset: A number in the range 0 through 65 535 that gives the position in bytes from the start of the record.

Element 2: Type of conversion

The type of conversion to be done on the field.

***SAME:** Keep the value currently specified in the CVT entry.

***CHAR:** The data is alphabetic and will be converted from the code page specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***PACKED:** The data is in packed decimal format and will not be converted.

***BINARY:** The data is in binary format and will not be converted.

***IDC:** The data is in ideographic format and will be converted from the character set specified in the **Character identifier (CNVCHRID)** parameter to the OS/400 code page specified in the OS/400 system values.

***INTEL:** The data is in Intel format and will be converted by reversing the bytes. The field can have a length of two or four only.

Element 3: Length of conversion

The length in bytes of the field to be converted.

***SAME:** Keep the value currently specified in the CVT entry.

length: Enter a number in the range 1 through 65 535. If the data is in Intel format, this field must contain 2 or 4.

Element 4: User-specified DBCS data

This element, which is valid only when ***CHAR** is specified for the field, indicates whether or not the data to be converted contains DBCS characters. DBCS data should be indicated by SO/SI delimiters. The DBCS data will not be converted.

***SAME:** Keep the value currently specified in the CVT entry.

***NO:** The data does not have SO/SI characters.

***YES:** The data has SO/SI characters.

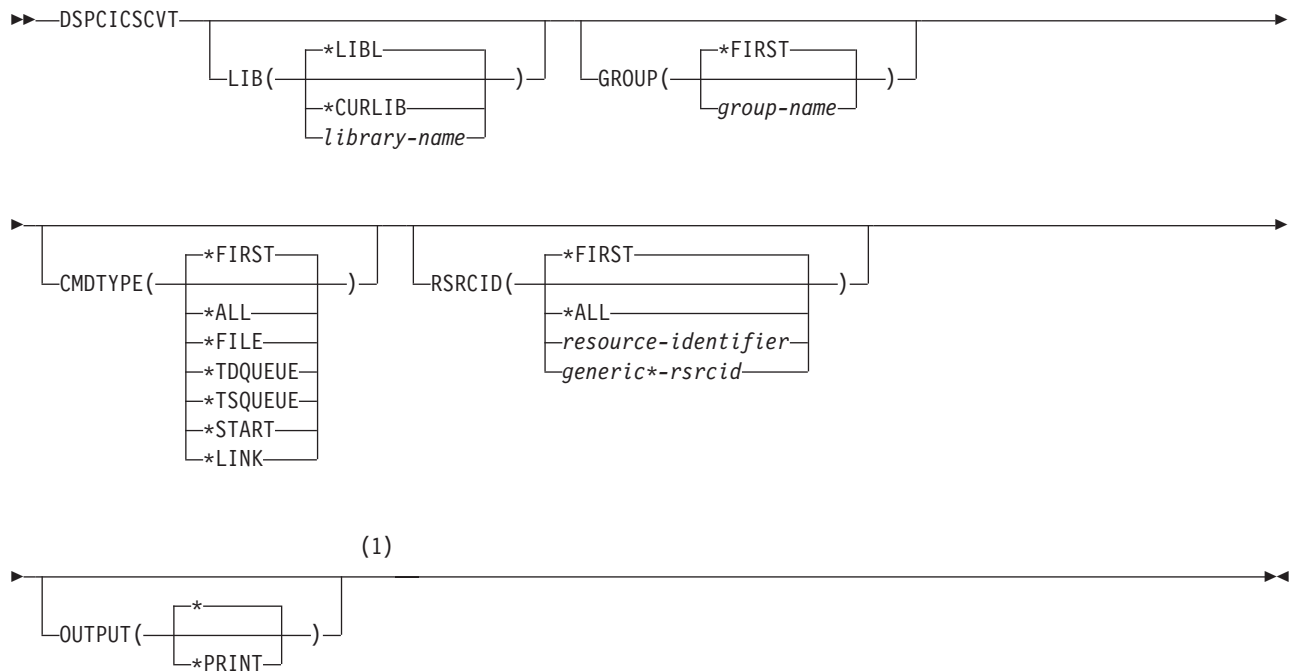
Examples

```
CHGCICSCVT LIB(PROD) GROUP(ACTPAY)
  CMDTYPE(*FILE)
  RSRCID(PAYABLES) CNVCHRID(1 1)
  CNVINFL((0 *CHAR 250 *NO))
```


This command changes the CVT entry called *FILEPAYABLES located in group ACTPAY in OS/400 library PROD. The conversion of the user data that is sent from, or to, another CICS system, uses the code page 1 and character set 1. It converts user data starting in position 0, for a length of 250, as character data with no SO/SI delimiters.

Using the DSPCICSCVT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Conversion Vector Table (DSPCICSCVT) command to display a CVT entry. You can only view this entry; you can neither make changes to it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the CVT entry to be displayed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

API command type (CMDTYPE)

Enter a command type. This parameter will be used with the **Resource identifier (RSRCID)** parameter to locate the required CVT entries.

Possible values are:

***FIRST:** Display the first CVT entry that matches the **Resource identifier (RSRCID)** parameter value.

***ALL:** Display all CVT entries that match the **Resource identifier (RSRCID)** parameter value.

***FILE:** Display all ***FILE** entries that match the **Resource identifier (RSRCID)** parameter value.

***TDQUEUE:** Display all ***TDQUEUE** entries that match the **Resource identifier (RSRCID)** parameter value.

***TSQUEUE:** Display all ***TSQUEUE** entries that match the **Resource identifier (RSRCID)** parameter value.

***START:** Display all ***START** entries that match the **Resource identifier (RSRCID)** parameter value.

***LINK:** Display all ***LINK** entries that match the **Resource identifier (RSRCID)** parameter value.

Resource identifier (RSRCID)

Enter a resource identifier. This parameter will be used with the **Command type (CMDTYPE)** parameter to locate the required CVT entries.

Possible values are:

***FIRST:** Display the first CVT entry that matches the **Command type (CMDTYPE)** parameter value.

***ALL:** Display all CVT entries that match the **Command type (CMDTYPE)** parameter value.

resource-identifier: The resource identifier may have a maximum length of 8 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

generic-rsrcid:* Specify the generic resource identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all CVT entries with a resource identifier beginning with the generic name, and matching the **Command type (CMDTYPE)** parameter value, are shown. If an asterisk is not included with the generic name, the system assumes the value to be the complete resource identifier.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT**: The output is printed with the job spool output.

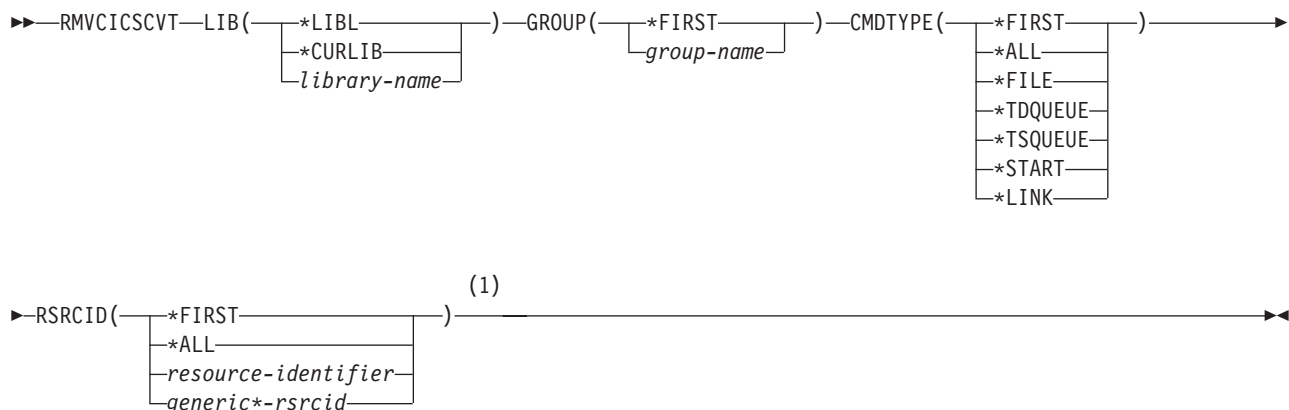
Examples

```
DSPCICSCVT LIB(PROD) GROUP(ACTPAY)
          CMDTYPE(*FILE) RSRCID(*ALL)
```

This command displays all CVT entries with a CMDTYPE of *FILE located in group ACTPAY in OS/400 library PROD.

Using the RMVCICSCVT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Conversion Vector Table (RMVCICSCVT) command to delete an entry from the CVT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL**: The library list is used to locate the first OS/400 library that contains the group.

***CURLIB**: The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the CVT entry to be removed.

Possible values are:

RMVICSCVT

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

API command type (CMDTYPE)

Enter a command type. This parameter will be used with the **Resource identifier (RSRCID)** parameter to locate the required CVT entries.

Possible values are:

***FIRST:** Remove the first CVT entry that matches the **Resource identifier (RSRCID)** parameter value.

***ALL:** Remove all CVT entries that match the **Resource identifier (RSRCID)** parameter value.

***FILE:** Remove all ***FILE** entries that match the **Resource identifier (RSRCID)** parameter value.

***TDQUEUE:** Remove all ***TDQUEUE** entries that match the **Resource identifier (RSRCID)** parameter value.

***TSQUEUE:** Remove all ***TSQUEUE** entries that match the **Resource identifier (RSRCID)** parameter value.

***START:** Remove all ***START** entries that match the **Resource identifier (RSRCID)** parameter value.

***LINK:** Remove all ***LINK** entries that match the **Resource identifier (RSRCID)** parameter value.

Resource identifier (RSRCID)

This parameter will be used with the **Command type (CMDTYPE)** parameter to locate the required CVT entries.

Possible values are:

***FIRST:** Remove the first CVT entry that matches the **Command type (CMDTYPE)** parameter value.

***ALL:** Remove all CVT entries that match the **Command type (CMDTYPE)** parameter value.

resource-identifier: The resource identifier may have a maximum length of 8 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

generic-rsrcid:* Specify the generic name of a resource identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all CVT entries with a resource identifier beginning with the generic name, and matching the **Command type (CMDTYPE)** parameter value, are removed. If an asterisk is not included with the generic name, the system assumes the value to be the complete resource identifier.

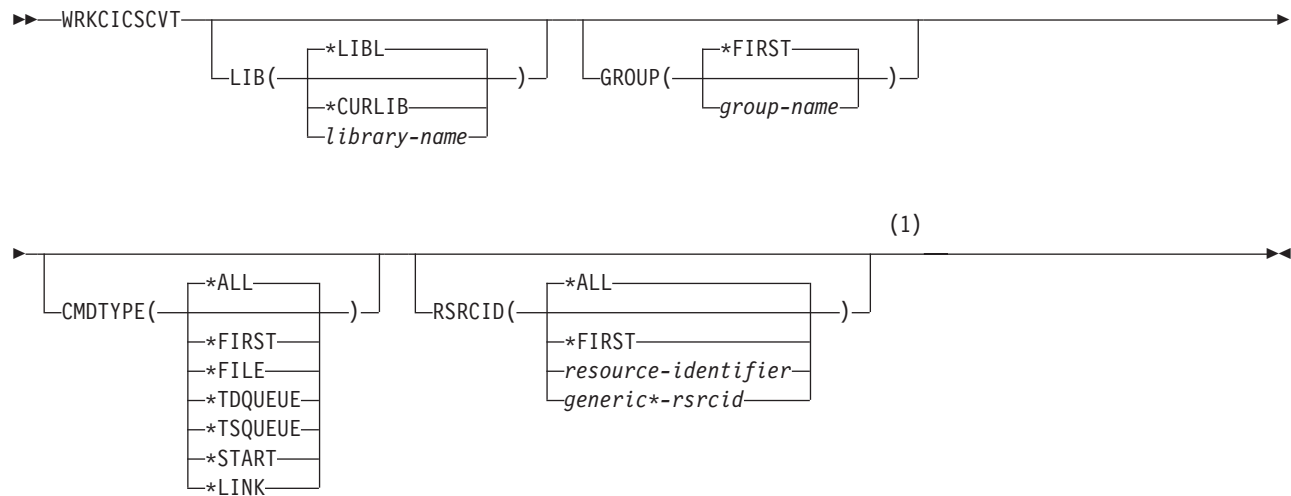
Examples

```
RMVICSCVT LIB(PROD) GROUP(ACTPAY)
          CMDTYPE(*FILE) RSRCID(PAY*)
```

This command removes all CVT entries, that start with *FILEPAY and end with anything, located in group ACTPAY in OS/400 library PROD.

Using the WRKCICSCVT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Conversion Vector Table (WRKCICSCVT) command to list entries in the CVT. You can then change, remove, copy, or display entries, or add new ones to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library which contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the CVT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

WRKCICSCVT

API command type (CMDTYPE)

Enter a command type. This parameter will be used with the **Resource identifier (RSRCID)** parameter to select the entries to be listed.

Possible values are:

- ***ALL**: List all the CVT entries that match the RSRCID parameter value.
- ***FIRST**: List the first CVT entry that matches the RSRCID parameter value.
- ***FILE**: List all file command entries.
- ***TDQUEUE**: List all TD queue entries.
- ***TSQUEUE**: List all TS queue entries.
- ***START**: List all start command entries.
- ***LINK**: List all link command entries.

Resource identifier (RSRCID)

Enter the resource identifier. This will be used with the **Command type (CMDTYPE)** parameter value to select the CVT entries to be listed.

Possible values are:

- ***ALL**: List all CVT entries that match the **Command type (CMDTYPE)** parameter value.
- ***FIRST**: List the first CVT entry that matches the **Command type (CMDTYPE)** parameter value.

resource-identifier: The resource identifier may have a maximum length of 8 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

generic-rsrcid*: Specify the generic name of the resource identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all CVT entries that begin with that generic name, and that match the **Command type (CMDTYPE)** parameter value, are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete resource identifier.

Examples

```
WRKCICSCVT LIB(PROD) GROUP(ACTPAY)
```

This command lists all CVT entries located in group ACTPAY in OS/400 library PROD.

Managing destination definitions

Each destination control table (DCT) entry defines a destination to which data may be sent. In CICS systems, a destination is a transient data (TD) queue. There are four types of destination:

- Intrapartition, which exists in the local CICS region
- Extrapartition, which exists outside the local CICS region
- Remote, which exists on another CICS system
- Indirect, which points to a destination controlled by another DCT entry

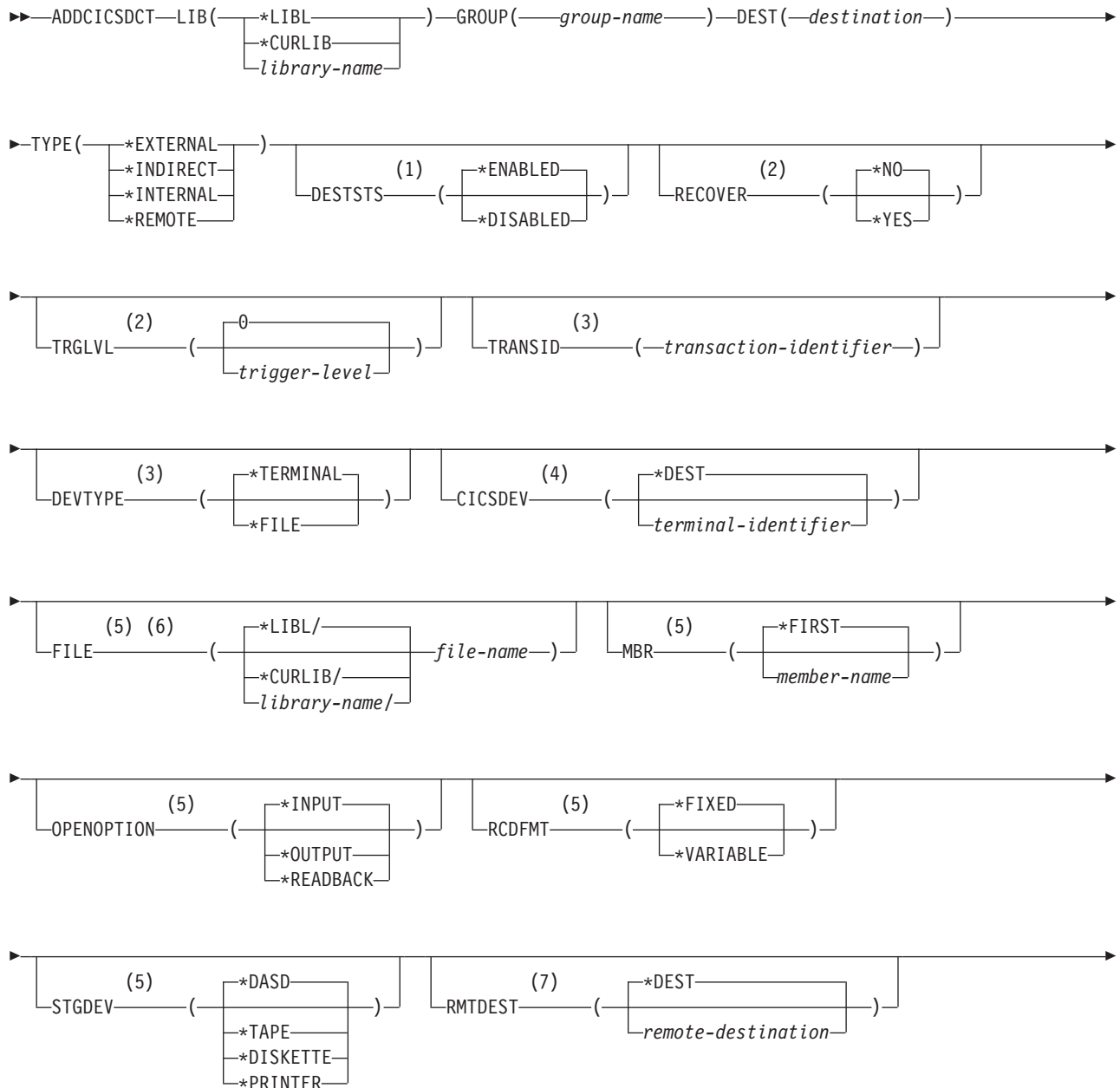
Create one DCT entry for each transient data queue.

Also use the DCT entry to define whether or not an intrapartition TD queue may be used for automatic transaction initiation (ATI).

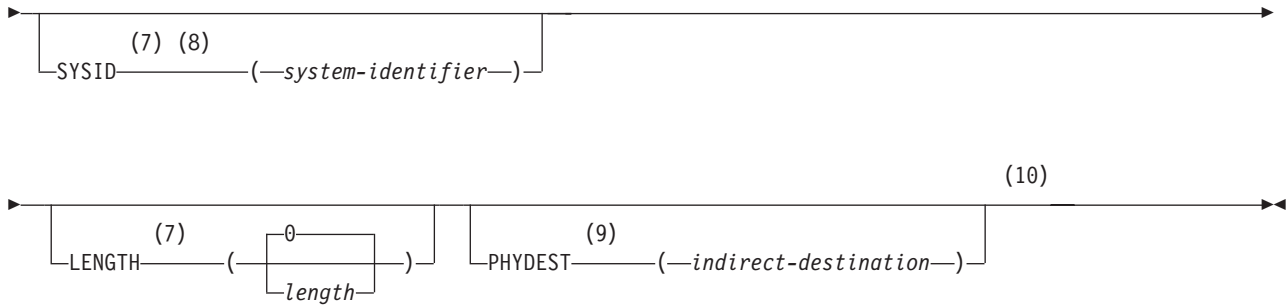
See “Defining temporary storage and transient data files” on page 63 and “Transient data considerations” on page 85 for guidance information.

Using the ADDCICSDCT command

Job: B,I Pgm: B,I REXX: B,I Exec



ADDCICSDCT



Notes:

- 1 The DESTSTS parameter is not valid when TYPE(*INDIRECT) is specified.
- 2 The RECOVER parameter and TRGLVL parameter are valid only when TYPE(*INTERNAL) is specified.
- 3 The TRANSID parameter and DEVTYPE parameter are not valid when TRGLVL(0) is specified.
- 4 The CICSDEV parameter is valid only when DEVTYPE(*TERMINAL) is specified.
- 5 The FILE parameter, MBR parameter, OPENOPTION parameter, RCDFMT parameter, and SSTGDEV parameter are valid only when TYPE(*EXTERNAL) is specified.
- 6 The FILE parameter is required when TYPE(*EXTERNAL) is specified.
- 7 The RMTDEST parameter, SYSID parameter, and LENGTH parameter are valid only when TYPE(*REMOTE) is specified.
- 8 The SYSID parameter is required when TYPE(*REMOTE) is specified.
- 9 The PHYDEST parameter is valid, and required, only when TYPE(*INDIRECT) is specified.
- 10 All parameters preceding this point can be specified positionally.

Function

Use the Add CICS/400 Destination Control Table (ADDCICSDCT) command to add an entry to the DCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this destination control table entry is to be added.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Destination (DEST)

Enter the destination identifier. This identifier will be used in EXEC CICS

commands to access transient data queues. This identifier is also the name that will be used to identify this DCT entry. If you are defining the CSMT log, this parameter must be CSMT.

destination: The destination identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Type (TYPE)

Enter the type of the transient data queue.

Possible values are:

***INTERNAL:** Indicates an intrapartition destination, that is a destination within the CICS/400 control region.

***INDIRECT:** Indicates an indirect destination, that is a logical destination that points to another destination, which is defined in the DCT as an internal, external, or remote destination. This allows several logical destinations to be merged into one physical destination.

***EXTERNAL:** Indicates an extrapartition destination, that is a destination outside, but allocated to, the CICS/400 control region.

***REMOTE:** Indicates a remote destination, that is a destination on another system or region, where it is defined in the DCT as an internal destination. This type is not valid if you are defining the CSMT log.

Optional parameters

Status (DESTSTS)

Indicates whether or not the destination identifier can be used. This parameter is not valid for indirect destinations, that is when the **Type (TYPE)** parameter contains *INDIRECT.

Possible values are:

***ENABLED:** This destination identifier can be used. If the **Type (TYPE)** parameter contains *EXTERNAL, then this transient data queue will be opened when the runtime system is installed.

***DISABLED:** This destination identifier cannot be used. If the **Type (TYPE)** parameter contains *EXTERNAL, then this transient data queue will not be opened when the runtime system is installed.

Recoverable (RECOVER)

Indicates whether or not the transient data queue associated with the destination identifier is recoverable. The parameter is valid only for an internal destination, that is when the **Type (TYPE)** parameter contains *INTERNAL.

Note: If this DCT entry is defining the CSMT log, this field value must be *NO.

Possible values are:

***NO:** The transient data queue is not recoverable.

***YES:** The transient data queue is recoverable.

ATI trigger level (TRGLVL)

Enter the number of records to accumulate in the queue in order to start automatically the transaction to process them. This is known as automatic transaction initiation (ATI). This field is valid only for internal destinations, that is when the **Type (TYPE)** parameter contains *INTERNAL.

The transaction identifier is specified in the transaction identifier field.

Possible values are:

0: There is no ATI for this destination.

trigger-level: Enter a number in the range 0 through 32 767.

Transaction (TRANSID)

Enter the name of the transaction to be started when the number of records in the queue reaches the trigger level. This parameter is not valid if the **Trigger level (TRGLVL)** parameter contains 0.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

CICS device indicator (DEVTYPE)

Indicates whether or not a terminal identifier is to be associated with the ATI transaction identifier. This parameter is not valid if the **Trigger level (TRGLVL)** parameter contains 0.

Possible values are:

***TERMINAL:** A terminal identifier will be associated with the transaction identifier.

***FILE:** No terminal identifier will be associated with the transaction identifier.

CICS device (CICSDEV)

Enter the identifier of the terminal to be used to run the ATI transaction. This parameter is valid only when the **CICS device indicator (DEVTYPE)** parameter contains *TERMINAL.

Possible values are:

***DEST:** The identifier entered in the destination field will be used as the terminal identifier.

terminal-identifier: The terminal identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

File (FILE)

Enter the name of the file that will be used by the destination identifier. This field is valid only for an extrapartition destination, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible library values are:

***LIBL:** The library list for the job that is associated with the control region is used to locate the file.

***CURLIB:** The current library for the job that is associated with the control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

file-name: Specify the name of the file.

Member (MBR)

Enter the name of the member that will be used by the destination identifier. This field is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

***FIRST**: No member is specified. The first member is used.

member-name: Specify the name of the member.

File processing (OPENOPTION)

Indicates how the file is to be processed, namely:

- A read-forward only input file
- A write-only output file
- A read-backward only input file

This parameter is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

***INPUT**: The file is to be used for input that will only be read forward.

***OUTPUT**: The file is used for output.

***READBACK**: The file is used for input that will only be read backward.

Record format (RCDFMT)

Indicates the record format of the file. This field is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

***FIXED**: All records in the file have the same length.

***VARIABLE**: The records in the file are of variable lengths.

Device (STGDEV)

Indicates the storage medium of the file. This parameter is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

- ***DASD**: The file is located on disk storage.
- ***TAPE**: The file is located on magnetic tape.
- ***DISKETTE**: The file is located on diskette.
- ***PRINTER**: The file is written to a printer.

For more information on creating destinations on these media, see page 86.

Remote destination (RMTDEST)

Enter the identifier by which the destination is known on the remote system. This field is valid only for remote destinations, that is when the **Type (TYPE)** parameter contains *REMOTE.

Possible values are:

***DEST**: The identifier entered in the destination identifier field will be used.

remote-destination: The destination identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

ADDCICSDCT

CICS system (SYSID)

Enter the remote system identifier. The system must have an entry in the remote TCS. This field is required only for a remote destination, that is when the **Type (TYPE)** parameter contains *REMOTE. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Maximum record length (LENGTH)

Enter the maximum length in bytes of the records in the transient data queue. This value must be the same as that specified for the queue in the remote system. This field is valid only for remote queues, that is when the type field contains *REMOTE.

Possible values are:

0: The LENGTH parameter will be specified in EXEC CICS READQ TD or WRITEQ TD commands within an application program.

length: Enter a number in the range 0 through 32 767.

Indirect destination (PHYDEST)

Enter the identifier of the destination to which this DCT entry points. This identifier should have a DCT entry defining the queue as *INTERNAL, *EXTERNAL, or *REMOTE. This field is valid only for indirect destinations, that is when the **Type (TYPE)** parameter contains *INDIRECT.

indirect-destination: The identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Examples

```
ADDCICSDCT LIB(CICSWORK) GROUP(ACCT)
  DEST(SAMA) TYPE(*INTERNAL)
  TRGLVL(55) TRANSID(ACCT04)
```

This command adds a DCT entry called SAMA to group ACCT in OS/400 library CICSWORK, with an internal queue type. The transaction ACCT04 will run when the number of records on the queue (the trigger level) reaches 55.

```
ADDCICSDCT LIB(CICSWORK) GROUP(ACCT)
  DEST(BETA) TYPE(*EXTERNAL)
  FILE(AACXTRA)
```

This command adds a DCT entry called BETA to group ACCT in OS/400 library CICSWORK, with an external queue type. The queue maps to physical file AACXTRA.

```
ADDCICSDCT LIB(CICSWORK) GROUP(ACCT)
  DEST(RMT1) TYPE(*REMOTE)
  SYSID(SYS1) LENGTH(512)
```

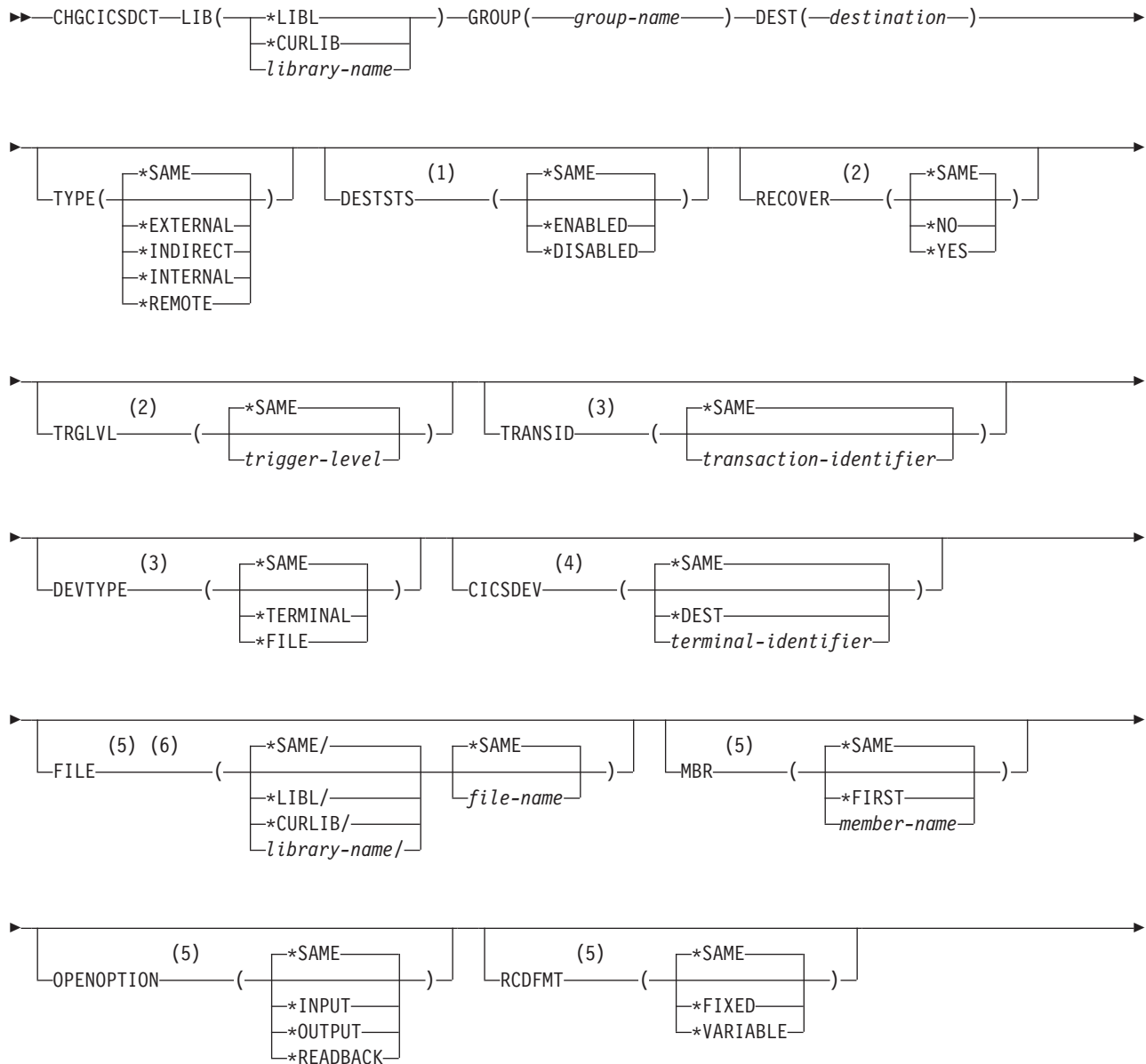
This command adds a DCT entry called RMT1 to group ACCT in OS/400 library CICSWORK, with a remote queue type. The remote system identifier is SYS1 and the remote record length is 512 bytes.

```
ADDCICSDCT LIB(CICSWORK) GROUP(ACCT)
  DEST(IND1) TYPE(*INDIRECT)
  PHYDEST(RMT1)
```

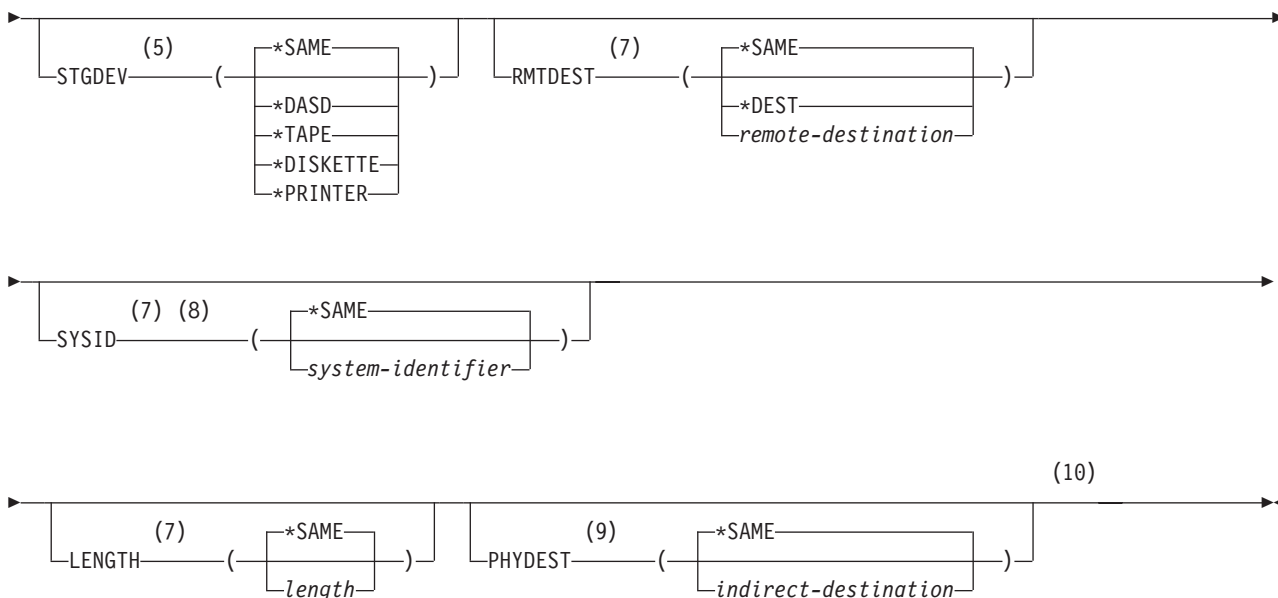
This command adds a DCT entry called IND1 to group ACCT in OS/400 library CICSWORK, with an indirect queue type. The indirect destination identifier is RMT1, which has already been defined in the DCT.

Using the CHGCICSDCT command

Job: B,I Pgm: B,I REXX: B,I Exec



CHGCICSDCT



Notes:

- 1 The DESTSTS parameter is not valid when TYPE(*INDIRECT) is specified.
- 2 The RECOVER parameter and TRGLVL parameter are valid only when TYPE(*INTERNAL) is specified.
- 3 The TRANSID parameter and DEVTYPE parameter are not valid when TRGLVL(0) is specified.
- 4 The CICSDEV parameter is valid only when DEVTYPE(*TERMINAL) is specified.
- 5 The FILE parameter, MBR parameter, OPENOPTION parameter, RCD_FMT parameter, and STGDEV parameter are valid only when TYPE(*EXTERNAL) is specified.
- 6 The FILE parameter is required when TYPE(*EXTERNAL) is specified.
- 7 The RMTDEST parameter, SYSID parameter, and LENGTH parameter are valid only when TYPE(*REMOTE) is specified.
- 8 The SYSID parameter is required when TYPE(*REMOTE) is specified.
- 9 The PHYDEST parameter is valid, and required, only when TYPE(*INDIRECT) is specified.
- 10 All parameters preceding this point can be specified positionally.

Function

Use the Change CICS/400 Destination Control Table (CHGCICSDCT) command to change a DCT entry.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the DCT entry to be changed.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Destination (DEST)

Enter the destination identifier. This identifier will be used in EXEC CICS commands to access transient data queues. This identifier is also the name that will be used to identify this DCT entry. If you are defining the CSMT log, this parameter must be CSMT.

destination: The destination identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters**Type (TYPE)**

Enter the type of transient data queue.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

***INTERNAL:** Indicates an intrapartition destination, that is a destination within the CICS/400 control region.

***INDIRECT:** Indicates an indirect destination, that is a logical destination that points to another destination, which is defined in the DCT as an internal, external, or remote destination. This allows several logical destinations to be merged into one physical destination.

***EXTERNAL:** Indicates an extrapartition destination, that is a destination outside, but allocated to, the CICS/400 control region.

***REMOTE:** Indicates a remote destination; that is a destination on another system or region. This type is not valid if you are defining the CSMT log.

Status (DESTSTS)

Indicates whether or not the destination identifier can be used. This field is not valid for indirect destinations, that is when the **Type (TYPE)** parameter contains *INDIRECT.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

***ENABLED:** This destination identifier can be used. If the **Type (TYPE)** parameter contains *EXTERNAL, then this transient data queue will be opened when the runtime system is installed.

***DISABLED:** This destination identifier cannot be used. If the **Type (TYPE)** parameter contains *EXTERNAL, then this transient data queue will not be opened when the runtime system is installed.

Recoverable (RECOVER)

Indicates whether or not the transient data queue associated with the destination identifier is recoverable or not recoverable. This parameter is valid only for an intrapartition destination, that is when the **Type (TYPE)** parameter contains *INTERNAL.

Note: If this DCT entry is defining the CSMT log, this field value must be *NO.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

***NO:** The transient data queue is not recoverable.

***YES:** The transient data queue is recoverable.

ATI trigger level (TRGLVL)

Enter the number of records to accumulate in the queue in order to start automatically the transaction to process them. This is known as automatic transaction initiation (ATI). This field is valid only for intrapartition destinations, that is when the **Type (TYPE)** parameter contains *INTERNAL.

The transaction identifier is specified in the transaction identifier field.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

trigger-level: Enter a number in the range 0 through 32 767.

Transaction (TRANSID)

Enter the name of the transaction to be started when the number of records reaches the trigger level. This parameter is not valid if the **Trigger level (TRGLVL)** parameter contains 0.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

CICS device indicator (DEVTYPE)

Indicates whether or not a terminal identifier is to be associated with the ATI transaction identifier. This parameter is not valid when the **Trigger level (TRGLVL)** parameter contains 0.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

***TERMINAL:** A terminal identifier is associated with the ATI transaction identifier.

***FILE:** No terminal identifier is associated with the ATI transaction identifier.

CICS device (CICSDEV)

Enter the identifier of the terminal to be used to run the ATI transaction. This parameter is valid only when the **CICS device indicator (DEVTYPE)** parameter contains *TERMINAL.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

***DEST:** The identifier entered in the destination field will be used as the terminal identifier.

terminal-identifier: The terminal identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

File (FILE)

Enter the name of the file that will be used by the destination identifier. This field is valid only for an extrapartition destination, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible library values are:

***SAME**: Keep the value currently specified in the DCT entry.

***LIBL**: The library list for the job that is associated with the control region is used to locate the file.

***CURLIB**: The current library for the job that is associated with the control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

Possible file name values are:

***SAME**: Keep the value currently specified in the DCT entry.

file-name: Specify the name of the file.

Member (MBR)

Enter the name of the member that will be used by the destination identifier. This field is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

***SAME**: Keep the value currently specified in the DCT entry.

***FIRST**: No member is specified. The first member is used.

member-name: Specify the name of the member.

File processing (OPENOPTION)

Indicates how the file is to be processed, namely:

- A read-forward only input file
- A write-only output file
- A read-backward only input file

This parameter is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

***SAME**: Keep the value currently specified in the DCT entry.

***INPUT**: The file is to be used for input that will only be read forward.

***OUTPUT**: The file is used for output.

***READBACK**: The file is used for input that will only be read backwards.

Record format (RCDFMT)

Indicates the record format of the file. This field is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

***SAME**: Keep the value currently specified in the DCT entry.

***FIXED**: All records in the file have the same length.

***VARIABLE:** The records in the file are of variable lengths.

Device (STGDEV)

Indicates the storage medium of the file. This field is valid only for extrapartition destinations, that is when the **Type (TYPE)** parameter contains *EXTERNAL.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

***DASD:** The file is located on disk storage.

***TAPE:** The file is located on magnetic tape.

***DISKETTE:** The file is located on diskette.

***PRINTER:** The file is written to a printer.

For more information on creating destinations on these media, see page 86.

Remote destination (RMTDEST)

Enter the identifier by which the destination is known on the remote system. This field is valid only for remote destinations, that is when the **Type (TYPE)** parameter contains *REMOTE.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

***DEST:** The identifier entered in the destination identifier field will be used.

remote-destination: The destination identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS system (SYSID)

Enter the remote system identifier. The system must have an entry in the remote TCS. This field is required only for remote destinations, that is when the **Type (TYPE)** parameter contains *REMOTE. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Maximum record length (LENGTH)

Enter the maximum length in bytes of the records in the transient data queue. This value must be the same as that specified for the queue in the remote system. This field is valid only for remote queues, that is when the type field contains *REMOTE. Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

0: The LENGTH parameter will be specified in EXEC CICS READQ TD or WRITEQ TD commands within an application program.

length: Enter a number in the range 0 through 32 767.

Indirect destination (PHYDEST)

Enter the identifier of the destination to which this DCT entry points. This identifier should have a DCT entry defining the queue as *INTERNAL,

*EXTERNAL, or *REMOTE. This field is valid only for indirect destinations, that is when the **Type (TYPE)** parameter contains *INDIRECT. Possible values are:

***SAME:** Keep the value currently specified in the DCT entry.

indirect-destination: The identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Examples

```
CHGCICSDCT LIB(CICSWORK) GROUP(ACCT)
           DEST(SAMA) TRGLVL(10)
```

This command changes the internal DCT entry called SAMA in group ACCT in OS/400 library CICSWORK. The transaction ACCT04 will now run when the number of records on the queue (the trigger level) reaches 10.

```
CHGCICSDCT LIB(CICSWORK) GROUP(ACCT)
           DEST(BETA) DESTSTS(*DISABLED)
```

This command changes the external DCT entry called BETA to group ACCT in OS/400 library CICSWORK. The queue has been disabled and is not available for use.

```
CHGCICSDCT LIB(CICSWORK) GROUP(ACCT)
           DEST(RMT1) RMTDEST(PRL1)
```

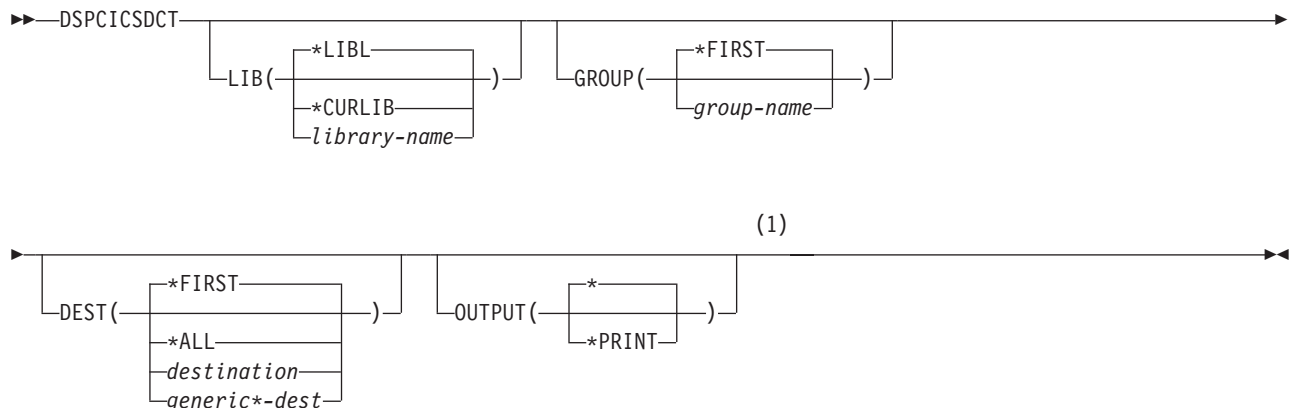
This command changes the remote DCT entry called RMT1 in group ACCT in OS/400 library CICSWORK. The remote destination identifier has been added to the definition.

```
ADDCICSDCT LIB(CICSWORK) GROUP(ACCT)
           DEST(IND1) PHYDEST(SAMA)
```

This command changes the indirect DCT entry called IND1 in group ACCT in OS/400 library CICSWORK. The indirect destination identifier is now SAMA, which has already been defined in the DCT as an internal destination.

Using the DSPCICSDCT command

Job: B,I Pgm: B,I REXX: B,I Exec



DSPCICSDCT

Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Destination Control Table (DSPCICSDCT) command to display a DCT entry. You can only view this entry; you can neither make changes to it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the DCT entry to be displayed.

Possible values are:

***FIRST:** No CICS/400 group is specified. The first CICS/400 group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Destination (DEST)

Enter the destination identifier of the DCT entry to be displayed.

Possible values are:

***FIRST:** Display the first DCT entry.

***ALL:** Display all the DCT entries.

destination: The destination identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-dest:* Specify the generic name of the destination identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all DCT entries with an identifier that begins with the generic name are displayed. If an asterisk is not included with the generic name, the system assumes the value to be the complete destination identifier.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*****: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT:** The output is printed with the job spool output.

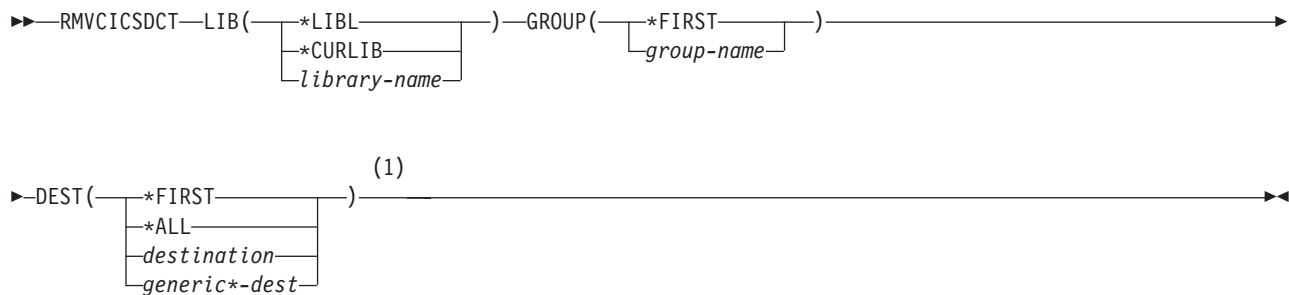
Examples

```
DSPCICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(*ALL)
```

This command displays all DCT entries located in group ACCT in OS/400 library CICSWORK.

Using the RMVCICSDCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Destination Control Table (RMVCICSDCT) command to delete an entry from the DCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the DCT entry to be removed.

Possible values are:

***FIRST:** No CICS/400 group is specified. The first CICS/400 group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Destination (DEST)

Enter the name of the DCT entry to be removed.

Possible values are:

***FIRST:** Remove the first DCT entry.

***ALL:** Remove all the DCT entries.

RMVICSDCT

destination: The maximum length is four characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-dest:* Specify the generic name of the destination identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all DCT entries with identifiers that begin with the generic name are removed. If an asterisk is not included with the generic name, the system assumes the value to be the complete destination identifier.

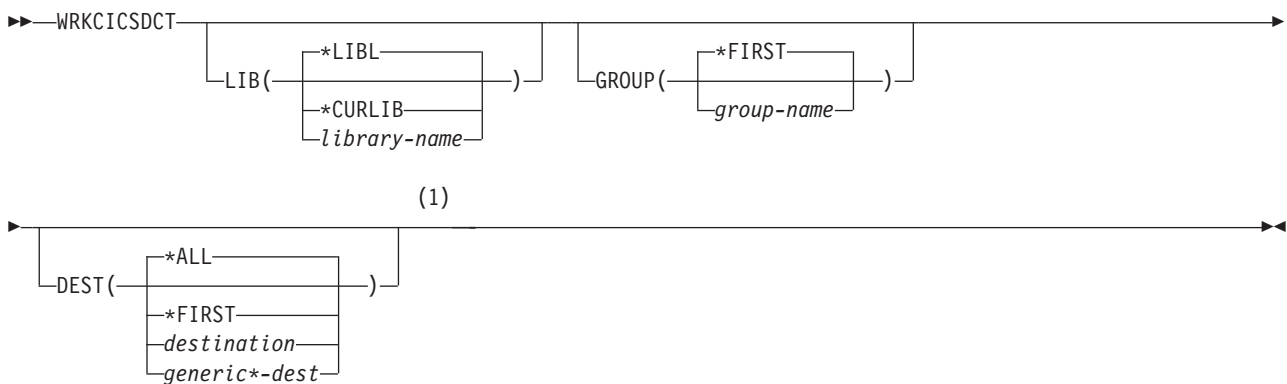
Examples

```
RMVICSDCT LIB(CICSWORK) GROUP(ACCT) DEST(ABC*)
```

This command removes all DCT entries that start with ABC and end with anything, located in group ACCT in OS/400 library CICSWORK.

Using the WRKCICSDCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Destination Control Table (WRKCICSDCT) command to list entries in the DCT. You can change, remove, copy, or display entries, or add new entries to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the DCT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Destination (DEST)

Enter the name of the DCT entries to be listed. This name is also the destination identifier that will be used in EXEC CICS transient data queue commands.

Possible values are:

***ALL:** List all DCT entries.

***FIRST:** List the first DCT entry.

destination: The destination may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-dest:* Specify the generic name of the destination identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all DCT entries that begin with the generic name are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete destination identifier.

Examples

```
WRKCICSDCT LIB(CICSWORK) GROUP(ACCT)
```

This command lists all DCT entries located in group ACCT in OS/400 library CICSWORK.

Managing file resource definitions

The file control table (FCT) defines all the files, both local and remote, that are used either by CICS/400 or by application programs. All files must be defined to have the following VSAM emulation file types:

- Key-sequence data set (KSDS), that is each record is accessed by predefined keys
- Entry-sequence data set (ESDS), that is each record is accessed by its relative byte address
- Relative-record data set (RRDS), that is each record is accessed by its record number

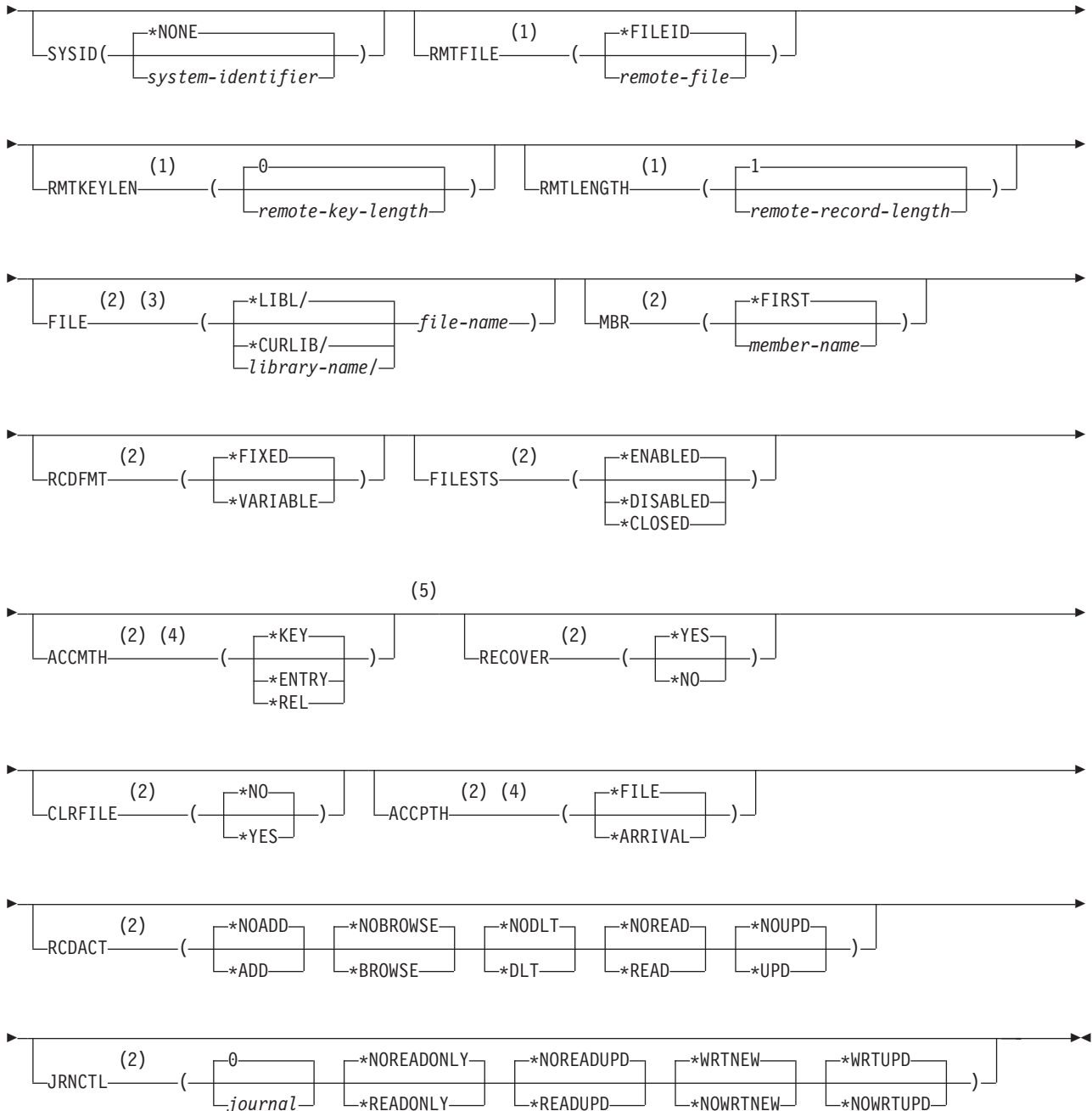
See “File control considerations” on page 88 for guidance information.

Using the ADDCICSFCT command

Job: B,I Pgm: B,I REXX: B,I Exec

```
▶▶—ADDCICSFCT—LIB(——*LIBL———)—GROUP(——group-name——)—FILEID(——file-identifier——)—▶▶
      |——*CURLIB——|
      |——library-name——|
```

ADDCICSFCT



Notes:

- 1 The RMTFILE parameter, RMTKEYLEN parameter, and RMTLENGTH parameter are not valid when SYSID(*NONE) is specified.
- 2 The FILE parameter, MBR parameter, RCDFMT parameter, FILESTS parameter, ACCMTH parameter, RECOVER parameter, CLRFILE parameter, ACCPTH parameter, RCDACT parameter, and JRNCTL parameter are valid only when SYSID(*NONE) is specified.
- 3 The FILE parameter is required when SYSID(*NONE) is specified.
- 4 ACCPTH(*ARRIVAL) is not valid when ACCMTH(*KEY) is specified.
- 5 All parameters preceding this point can be specified positionally.

Function

Use the Add CICS/400 File Control Table (ADDCICSFCT) command to add an entry to the file control table (FCT).

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible library values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this FCT entry is to be added.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS file (FILEID)

Enter the name by which this file is known. This is the name that will be used in EXEC CICS file control commands. This name is also used to identify this FCT entry.

file-identifier: The file identifier may have a maximum length of 8 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

CICS system (SYSID)

Enter the identifier of the system where the file is located. The system identifier must have a terminal control system (TCS) table entry. This parameter is only required for remote files. If this parameter contains the ID of the local system, the entry will be treated as if *NONE had been entered.

Possible values are:

***NONE:** The file is held on the local system.

system-identifier: A system ID up to four characters in length. The first character must be alphabetic, or one of the special characters, \$, @, or #.

The remaining characters can be alphanumeric or \$, @, or #.

Remote CICS file (RMTFILE)

Enter the identifier by which the file is known on the remote system. This parameter is not valid for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***FILEID:** The name entered in the **File (FILE)** parameter will be used as the remote file name.

remote-file: A file identifier up to 8 characters in length. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote maximum key length (RMTKEYLEN)

Enter the length in bytes of the key field for the file. This parameter is valid for key-sequenced data sets only. The value must be the same as that specified in the FCT entry on the remote system.

This parameter is not valid for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

0: File does not have a key field. This is the value that should be entered for entry-sequenced data sets and relative record data sets.

remote-key-length: A number in the range 1 through 32 767.

Remote maximum record length (RMTLENGTH)

Enter the length in bytes of the maximum record size for the file. The value specified must correspond to the size of the record associated with the file, in the remote system. This parameter is not valid for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

1: File has a record length of one.

remote-record-length: A number in the range 1 through 32 767.

File (FILE)

Enter the library and file name of the file that will be utilized by this file identifier. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

Specify one of the following library value:

***LIBL:** The library list for the job that is associated with the CICS/400 control region is used to locate the file.

***CURLIB:** The current library for the job that is associated with the CICS/400 control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

file-name: Specify the name of the file.

Member (MBR)

Enter the name of the member in the file that will be utilized by the file identifier. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***FIRST:** No file member is specified. The first member in the file is used.

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

Record format (RCDFMT)

Indicates whether the record format of the file is fixed length or variable length. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***FIXED:** All records in the file have the same length.

***VARIABLE:** The records in the file are of variable lengths.

Status (FILESTS)

Indicates whether or not the file associated with the file identifier can be used. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***ENABLED:** The file can be used.

***DISABLED:** The file cannot be used until it has been enabled.

***CLOSED:** The file cannot be used until it has been opened.

Access method (ACCMTH)

Enter the method that is to be used to access the file. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***KEY:** The records in the file will be accessed by key. The file must be a KSDS.

***ENTRY:** The records in the file will be accessed by entry-sequence number. The file must be an ESDS.

***REL:** The records in the file will be accessed by relative-record number. The file must be an RRDS.

Note that this parameter is ignored if you enter *ARRIVAL in the **Access path (ACCPH)** parameter.

Recoverable (RECOVER)

Indicates whether or not the file associated with the file identifier is recoverable. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***YES:** The file is recoverable.

***NO:** The file is not recoverable.

Clear file (CLRFILE)

Indicates whether or not the file is to be cleared when this file identifier is first opened. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***NO:** Do not clear the file when opened.

***YES:** Clear the file when opened.

Access path (ACCPH)

Indicates whether the file associated with the file identifier should be processed in arrival sequence (sequential) or in accordance with the value in the **Access method (ACCMTH)** parameter (that is by key, entry-sequence number, or relative-record number). This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***FILE:** The file will be processed in accordance with the value in the **Access method (ACCMTH)** parameter.

***ARRIVAL:** The file will be processed in the arrival sequence (sequentially), regardless of the value in the **Access method (ACCMTH)** parameter.

File processing (RCDACT)

Indicates whether or not records in the file associated with the file identifier can be added, browsed, deleted, read, or updated. This parameter is not valid for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

There are 5 elements to this field. Possible values are:

Element 1: Add allowed

Indicates whether or not records can be added to the file.

*NOADD: Records cannot be added.

*ADD: Records can be added.

Element 2: Browse allowed

Indicates whether or not records can be browsed.

*NOBROWSE: Records cannot be browsed.

*BROWSE: Records can be browsed.

Element 3: Delete allowed

Indicates whether or not records can be deleted.

*NODLT: Records cannot be deleted.

*DLT: Records can be deleted.

Element 4: Read allowed

Indicates whether or not records can be read.

*NOREAD: Records cannot be read. (This is the default, unless either the browse or update capability is specified.)

*READ: Records can be read.

Element 5: Update allowed

Indicates whether or not records can be updated.

*NOUPD: Records cannot be updated.

*UPD: Records can be updated.

Journal processing (JRNCTL)

Specifies the number of the journal file to be used with this file, and which READ and WRITE operations are to be journaled. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

There are 5 elements to this field. Possible values are:

Element 1: Number

The journal number to be used for recording READ and WRITE activity on this file. The journal number must be associated with a file through a journal control table (JCT) entry.

0: No journaling is to occur for the file associated with this file identifier.
journal-number: A number in the range 1 through 99.

Element 2: READ ONLY operations

Indicates whether or not READ ONLY operations will be journaled.

*NOREADONLY: READ ONLY operations will not be journaled.

*READONLY: READ ONLY operations will be journaled.

Element 3: READ UPDATE operations

Indicates whether or not READ UPDATE operations will be journaled.

*NOREADUPD: READ UPDATE operations will not be journaled.

*READUPD: READ UPDATE operations will be journaled.

Element 4: WRITE NEW operations

Indicates whether or not WRITE NEW operations will be journaled.

*WRTNEW: WRITE NEW operations will be journaled

*NOWRTNEW: WRITE NEW operations will not be journaled.

Element 5: WRITE UPDATE operations

Indicates whether or not WRITE UPDATE operations will be journaled.

*WRTUPD: WRITE UPDATE operations will be journaled.

*NOWRTUPD: WRITE UPDATE operations will not be journaled.

Examples

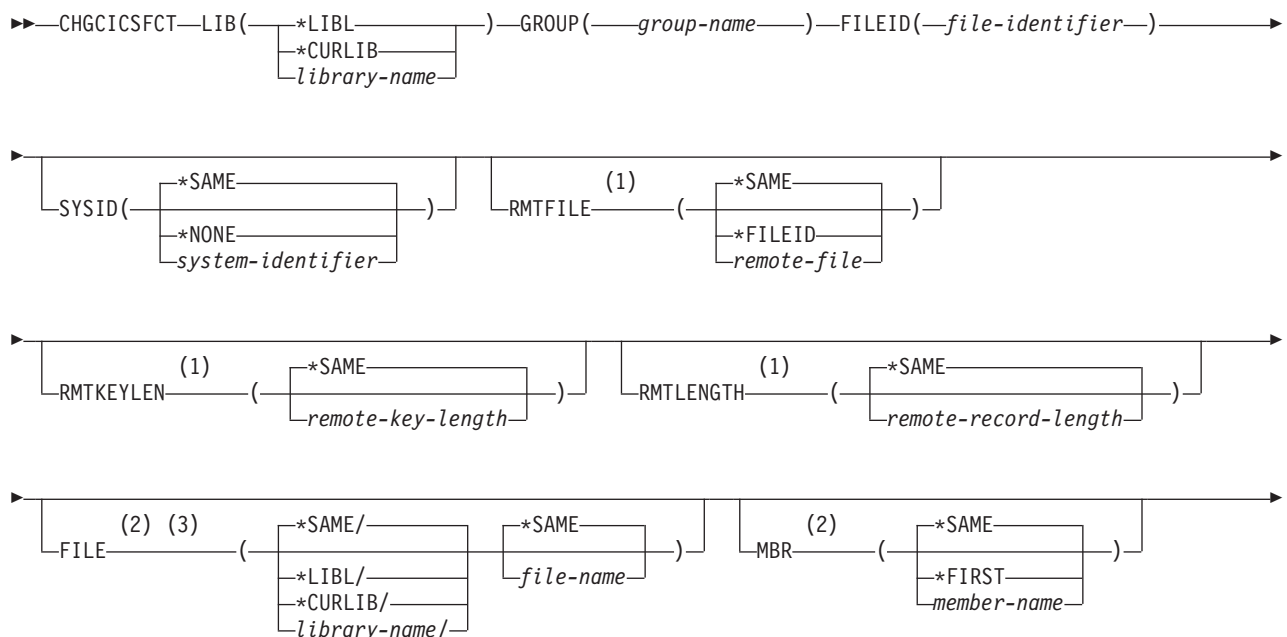
```
ADDCICSFCT LIB(CICSWORK) GROUP(ACCT)
FILEID(ACCTFIL)
FILE(CICSWORK/ACCTFIL) RECOVER(*NO)
RCDACT(*ADD *NOBROWSE *DLT *READ *UPD)
JRNCTL(2 *NOREADONLY *NOREADUPD)
```

This command adds an FCT entry called ACCTFIL to group ACCT in OS/400 library CICSWORK. The **Access method (ACCMTH)** parameter has been omitted, indicating, by default, a key-sequenced data set (KSDS). The file is not recoverable. READ ONLY and READ UPDATE activity will not be journaled. By default, WRITE NEW and WRITE UPDATE operations will be journaled.

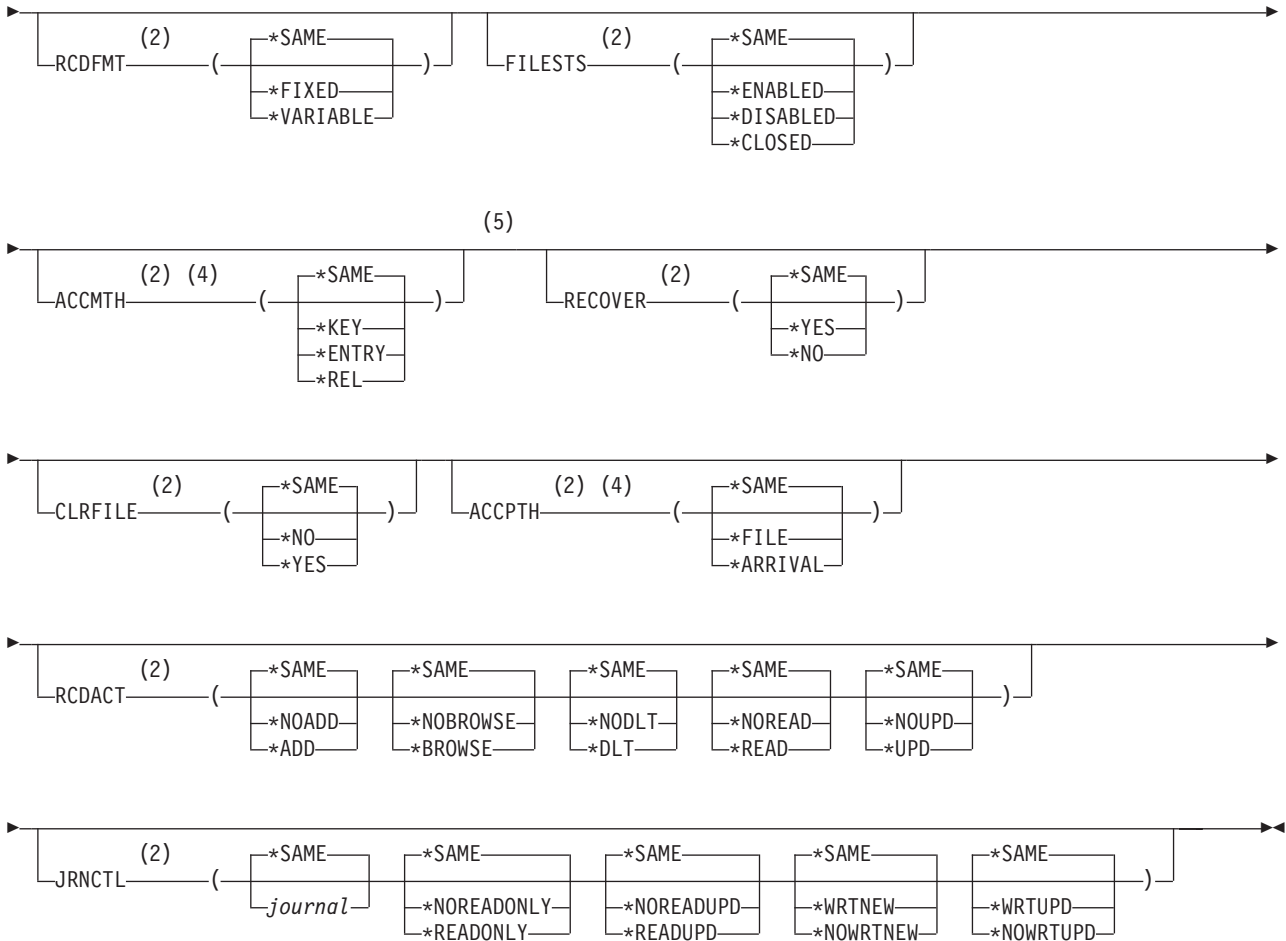
Examples of FCT definitions for EDSO, RRDS and alternate index files are given in "Example FCT definitions" on page 90.

Using the CHGCICSFCT command

Job: B,I Pgm: B,I REXX: B,I Exec



CHGCICSFCT



Notes:

- 1 The RMTFILE parameter, RMTKEYLEN parameter, and RMTLENGTH parameter are not valid when SYSID(*NONE) is specified.
- 2 The FILE parameter, MBR parameter, RCDFMT parameter, FILESTS parameter, ACCMTH parameter, RECOVER parameter, CLRFILE parameter, ACCPTH parameter, RCDACT parameter, and JRNCTL parameter are valid only when SYSID(*NONE) is specified.
- 3 The FILE parameter is required when SYSID(*NONE) is specified.
- 4 ACCPTH(*ARRIVAL) is not valid when ACCMTH(*KEY) is specified.
- 5 All parameters preceding this point can be specified positionally.

Function

Use the Change CICS/400 File Control Table (CHGCICSFCT) command to change an entry in the FCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the FCT entry to be changed.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS file (FILEID)

Enter the name by which this file is known. This is the name that will be used in EXEC CICS file control commands. This name is also used to identify this FCT entry.

file-identifier: The file identifier may have a maximum length of 8 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

CICS system (SYSID)

Enter the identifier of the system where the file is located. The system identifier must have a local terminal control system (TCS) table entry. This parameter is required only for remote files. If this parameter contains the identifier of the local system, this entry will be treated as if *NONE had been entered.

Possible values are:

***SAME:** Keep the system ID currently specified in the FCT entry.

***NONE:** The file is held on the local system.

system-identifier: A system ID up to 4 characters in length. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote CICS file (RMFILE)

Enter the identifier by which the file is known on the remote system. This parameter is not valid for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in this FCT entry.

***FILEID:** The name entered in the **File (FILE)** parameter will be used as the remote file name.

remote-file: A file identifier up to 8 characters in length. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote maximum key length (RMTKEYLEN)

Enter the length in bytes of the key field for the file. This parameter is valid for key-sequenced data sets only. The value must be the same as that specified in the FCT entry in the remote system.

This parameter is not valid for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep to value currently specified in the FCT entry.

remote-key-length: A number in the range 0 through 32 767. Zero (0) should be specified for entry-sequenced and relative-record data sets.

Remote maximum record length (RMTLENGTH)

Enter the length in bytes of the maximum record size for the file. The value specified must correspond to the size of the record associated with the file in the remote system. This parameter is not valid for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME**: Keep the value currently specified in this FCT entry.

remote-record-length: A number in the range 1 through 32 767.

File (FILE)

Enter the library and file name of the file that will be utilized by this file identifier. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

Specify one of the following library values:

***SAME**: Keep the value currently specified in this FCT entry.

***LIBL**: The library list for the job that is associated with the CICS/400 control region is used to locate the file.

***CURLIB**: The current library for the job that is associated with the CICS/400 control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

Specify one of the following name values:

***SAME**: Keep the name currently specified in this FCT entry.

file-name: Specify the name of the file.

Member (MBR)

Enter the name of the member in the file that will be utilized by this file identifier. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME**: Keep the value currently specified in this FCT entry.

***FIRST**: No file member is specified. The first member in the file is used.

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

Record format (RCDFMT)

Indicates whether the record format of the file is fixed length or variable length. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME**: Keep the value currently specified in this FCT entry.

***FIXED**: All records in the file have the same length.

***VARIABLE**: The records in the file are of variable lengths.

Status (FILESTS)

Indicates whether or not the file associated with the file identifier can be used. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in this FCT entry.

***ENABLED:** The file can be used.

***DISABLED:** The file cannot be used until it has been enabled.

***CLOSED:** The file cannot be used until it has been opened.

Access method (ACCMTH)

Enter the method that is to be used to access the file. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in this FCT entry.

***KEY:** The records in the file will be accessed by key. The file must be a KSDS.

***ENTRY:** The records in the file will be accessed by entry-sequence number. The file must be an ESDS.

***REL:** The records in the file will be accessed by relative-record number. The file must be an RRDS.

Note that this parameter is ignored if you enter *ARRIVAL in the **Access path (ACCPH)** parameter.

Recoverable (RECOVER)

Indicates whether or not the file associated with the CICS/400 file identifier is recoverable. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in this FCT entry.

***YES:** The file is recoverable.

***NO:** The file is not recoverable.

Clear file (CLRFILE)

Indicates whether or not the file is to be cleared when the file associated with this file identifier is first opened. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in this FCT entry.

***NO:** Do not clear the file when opened.

***YES:** Clear the file when opened.

Access path (ACCPH)

Indicates whether the file associated with the file identifier should be processed in arrival sequence (sequential) or in accordance with the value in the **Access method (ACCMTH)** parameter (that is by key, entry-sequence number, or relative-record number). This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in this FCT entry.

***FILE:** The file will be processed in accordance with the value in the **Access method (ACCMTH)** parameter.

***ARRIVAL:** The file will be processed in the arrival sequence (sequentially), regardless of the value in the **Access method (ACCMTH)** parameter.

File processing (RCDACT)

Indicates whether or not records in the file associated with the file identifier can be added, browsed, deleted, read, or updated. This parameter is only valid for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

There are 5 elements to this field. Possible values are:

Element 1: Add allowed

Indicates whether or not records can be added to the file.

*SAME: Keep the value currently specified in this FCT entry.

*NOADD: Records cannot be added.

*ADD: Records can be added.

Element 2: Browse allowed

Indicates whether or not records can be browsed in the file.

*SAME: Keep the value currently specified in this FCT entry.

*NOBROWSE: Records cannot be browsed.

*BROWSE: Records can be browsed.

Element 3: Delete allowed

Indicates whether or not records can be deleted in the file.

*SAME: Keep the value currently specified in this FCT entry.

*NODLT: Records cannot be deleted.

*DLT: Records can be deleted.

Element 4: Read allowed

Indicates whether or not records can be read in the file.

*SAME: Keep the value currently specified in this FCT entry.

*NOREAD: Records cannot be read. (This is the default, unless either the browse or update ability is specified.)

*READ: Records can be read.

Element 5: Update allowed

Indicates whether or not records can be updated in the file.

*SAME: Keep the value currently specified in this FCT entry.

*NOUPD: Records cannot be updated.

*UPD: Records can be updated.

Journal processing (JRNCTL)

Specifies the number of the journal file to be used with this file, and which READ and WRITE operations are to be journaled. This parameter is valid only for a local file, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

Element 1: Number

The journal number to be used for recording READ and WRITE activity on this file. The journal number must be associated with a file through a journal control table (JCT) entry.

*SAME: Keep the value currently specified in this FCT entry.

journal: A journal number in the range 1 through 99. If the value specified is 0, then no journaling is to occur for the file associated with this file identifier.

Element 2: READ ONLY operations

Indicates whether or not READ ONLY operations will be journaled.

***SAME:** Keep the value currently specified in this FCT entry.

***NOREADONLY:** READ ONLY operations will not be journaled

***READONLY:** READ ONLY operations will be journaled.

Element 3: READ UPDATE operations

Indicates whether or not READ UPDATE operations will be journaled.

***SAME:** Keep the value currently specified in this FCT entry.

***NOREADUPD:** READ UPDATE operations will not be journaled.

***READUPD:** READ UPDATE operations will be journaled.

Element 4: WRITE NEW operations

Indicates whether or not WRITE NEW operations will be journaled.

***SAME:** Keep the value currently specified in this FCT entry.

***WRTNEW:** WRITE NEW operations will be journaled.

***NOWRTNEW:** WRITE NEW operations will not be journaled.

Element 5: WRITE UPDATE operations

Indicates whether or not WRITE UPDATE operations will be journaled.

***SAME:** Keep the value currently specified in this FCT entry.

***WRTUPD:** WRITE UPDATE operations will be journaled.

***NOWRTUPD:** WRITE UPDATE operations will not be journaled.

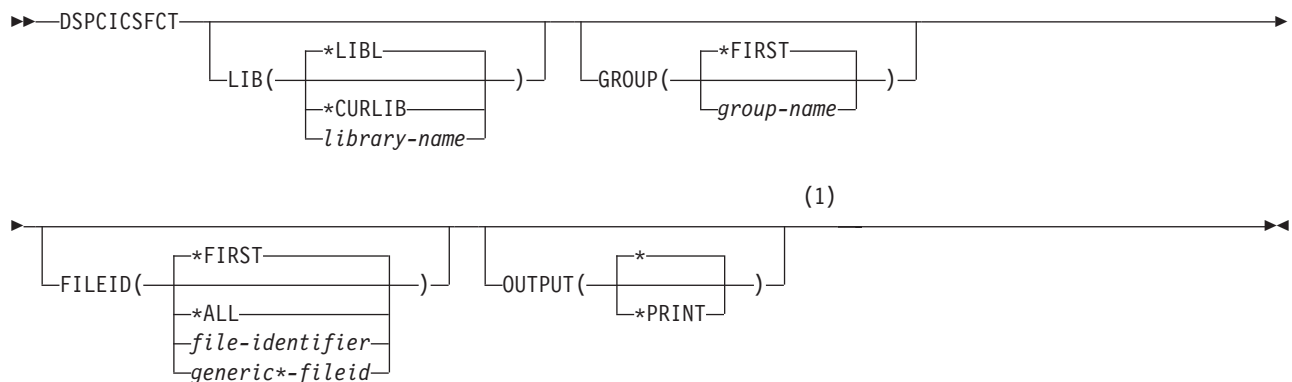
Examples

```
CHGCICSFCT LIB(CICSWORK) GROUP(ACCT)
FILEID(ACCTFIL) RECOVER(*YES)
JRNLCT(2 *READONLY)
```

This command changes the FCT entry called ACCTFIL in group ACCT in OS/400 library CICSWORK. The file is now recoverable and READ ONLY operations will be journaled.

Using the DSPCICSFCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 File Control Table (DSPCICSFCT) command to display an FCT entry. You can only view this entry; you can neither change it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the required FCT entry.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS file (FILEID)

Enter the name of the FCT entry to be displayed.

Possible values are:

***FIRST:** Display the first FCT entry.

***ALL:** Display all the FCT entries.

file-identifier: The file identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-fileid:* Specify the generic name of a file identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all FCT entries beginning with the generic name are displayed. If an asterisk is not included with the generic name, the system assumes the value to be the complete file identifier.

Location of output (OUTPUT)

Enter the location of the output from the DSPCICSFCT command.

Possible values are:

*****: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT:** The output is printed with the job spool output.

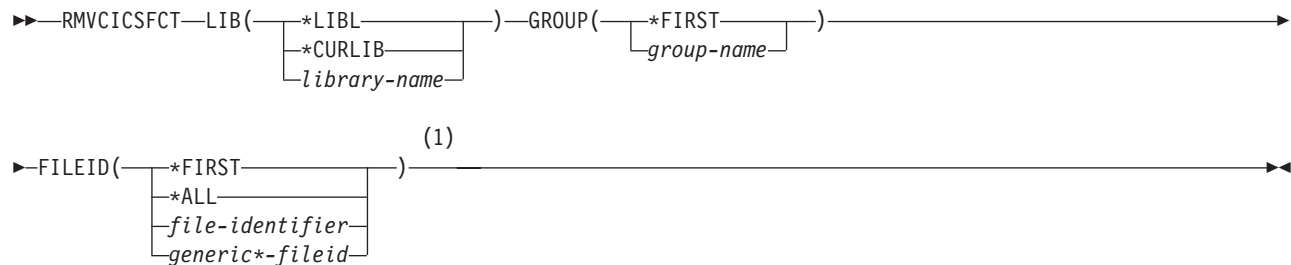
Examples

```
DSPCICSFCT LIB(CICSWORK) GROUP(ACCT) FILEID(*ALL)
```

This command displays all FCT entries located in group ACCT in OS/400 library CICSWORK.

Using the RMVCICSFCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 File Control Table (RMVCICSFCT) command to delete an entry from the FCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the FCT entry to be deleted.

Possible values are:

***FIRST:** No CICS/400 group is specified. The first CICS/400 group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS file (FILEID)

Enter the name of the FCT to be removed.

Possible values are:

***FIRST:** Remove the first FCT entry.

***ALL:** Remove all the FCT entries.

file-identifier: The file identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-fileid:* A generic name that identifies a number of FCT entries. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all FCT entries with identifiers beginning with the generic name will be removed. If an

RMVCSFCT

asterisk is not included with the generic name, the system assumes the value to be the complete file identifier.

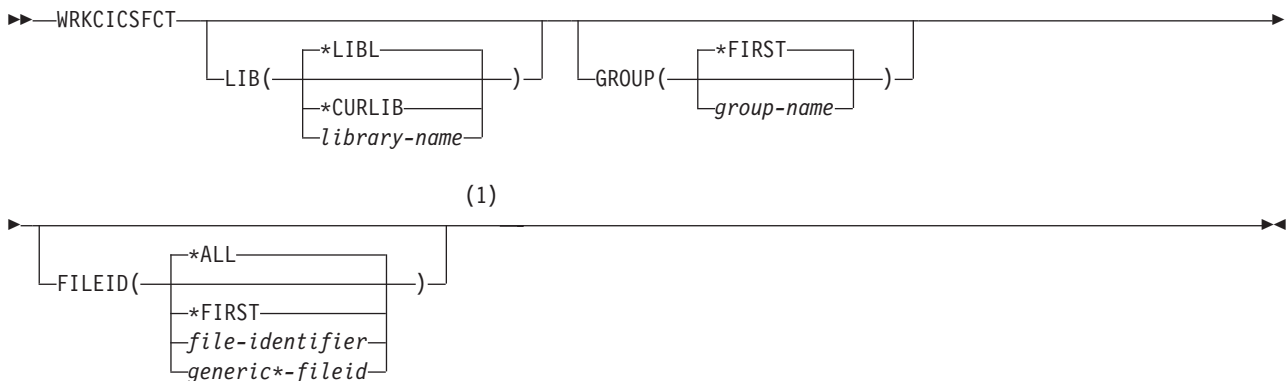
Examples

```
RMVCSFCT LIB(CICSWORK) GROUP(ACCT)
FILEID(ABC*)
```

This command removes all FCT entries that start with ABC and end with anything, located in group ACCT in OS/400 library CICSWORK.

Using the WRKCSFCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 File Control Table (WRKCSFCT) command to list entries in the FCT. You can change, remove, copy, or display entries, or add new entries to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the FCT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS file (FILEID)

Enter the name of the FCT entry to be listed. This is also the name used to identify the file in EXEC CICS commands.

Possible values are:

***ALL**: List all the FCT entries.

***FIRST**: List the first FCT entry.

file-identifier: The file identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-fileid*: Specify the generic name of the file identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all FCT entries beginning with the generic name are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete file identifier.

Examples

```
WRKCICSFCT LIB(CICSWORK) GROUP(ACCT)
```

This command lists all FCT entries located in group ACCT in OS/400 library CICSWORK.

Managing group list resource definitions

Each resource definition must belong to a group and each group must belong to a group list. The group list table (GLT) must itself be assigned to a group. The library and group of the GLT is specified in the SIT and will be used at system startup to select the required resource definitions.

You can create more than one GLT, but each GLT must be assigned to a different group. Only the GLT named in the SIT will be used at system startup. At startup, each entry in the GLT is read and the resources installed from those groups, excluding the GLT and the SIT. If you want to install any resources belonging to the group named in the GLTGRP parameter of the SIT, then you must add the name of that group to its own GLT. See “Specifying which groups are used” on page 47 for details and an example.

Each GLT entry specifies:

- The library name and group name of the GLT
- The library name and group name of a group to be installed at system startup

ADDCICSGLT

Using the ADDCICSGLT command

Job: B,I Pgm: B,I REXX: B,I Exec

```
▶▶—ADDCICSGLT—LIB(—*LIBL—)—GROUP(—group-name—)—INSLIB(—*LIBL—)——▶
      |—*CURLIB—|
      |—library-name—|
(1)
▶—INSGRP(—group-name—)——▶
```

Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Add CICS/400 Group List Table (ADDCICSGLT) command to add an entry to the GLT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which the GLT entry is to be added.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Install library (INSLIB)

The name of the OS/400 library that contains the group to be used when starting the control region. This is also known as the first part of the name of the GLT entry.

Note: When INSLIB(QCICS) is specified, then the INSGRP must be an IBM CICS/400 group.

Possible values are:

***LIBL:** The library list for the job that is associated with the control region is used to locate the group.

***CURLIB:** The current library for the job that is associated with the control region is used to locate the group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the group is located.

Install group (INSGRP)

The group to be used when starting the control region, that contains the tables. This is also known as the second part of the name of the GLT entry.

Note: When INSLIB(QCICS) is specified, then this must be one of the IBM groups.

The following describes which group is used to install which optional supplied transactions:

Group	Supplied Transaction
AEGCLI	CCIN - CICS Client INstall CTIN - CICS Terminal INstall
AEGEDF	CEBR - Browse Temporary Storage Queues CEDF - Execution Diagnostic Facility
AEGINTER	CECI - Command-Level Interpreter Facility CECS - Command-Level Syntax-Checking Facility
AEGISC	CMPX - Local Queuing Shipped CPMI - Mirror Transaction CRSR - Relay Transaction CRTE - Transaction Routing
AEGOPER	CEMT - Master Terminal Facility
AEGSPI	CEDA - Resource Definition Online Facility

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

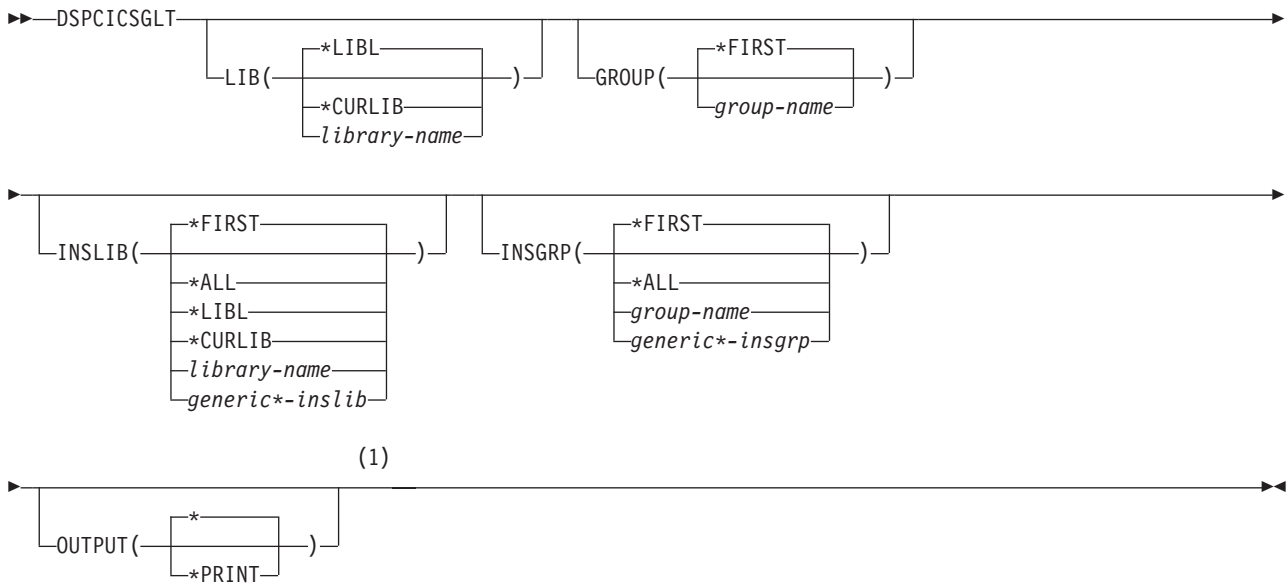
Examples

```
ADDCICSGLT LIB(SAMPLE1) GROUP(ACCT)
  INSLIB(QCICS) INSGRP(AEGEDF)
```

This command adds a GLT entry called QCICS AEGEDF to group ACCT in OS/400 library SAMPLE1.

Using the DSPCICSGLT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Group List Table (DSPCICSGLT) command to display a GLT entry. You can only view this entry; you can neither make changes to it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the GLT entry to be displayed.

Possible values are:

***FIRST:** No group is specified, the first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Install library (INSLIB)

Enter the name of the OS/400 library that contains the group to be listed.

Possible values are:

***FIRST:** Display the first GLT entry that matches the INSGRP parameter value.

***ALL:** Display all of the GLT entries that match the INSGRP parameter value.

***LIBL:** Display all of the GLT entries that have *LIBL specified and that match the INSGRP parameter value.

***CURLIB:** Display all of the GLT entries that have *CURLIB specified and that match the INSGRP parameter value.

library-name: The library name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-inslib:* Specify the generic name of the OS/400 library. A generic name is a string of one or more characters followed by an (*); for example, ABC*. If a generic name is specified, then all GLT entries with an OS/400 library name beginning with the generic name, and matching the INSGRP parameter value, are shown. If an asterisk is not included with the generic name, the system assumes the value to be the complete OS/400 library name.

Install group (INSGRP)

Enter the name of the group to be listed.

Possible values are:

***FIRST:** Display the first GLT entry that matches the INSLIB parameter value.

***ALL:** Display all of the GLT entries that match the INSLIB parameter value.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-insgrp:* Specify the generic name of the group. A generic name is a string of one or more characters followed by an asterisk(*); for example, ABC*. If a generic name is specified, then all GLT entries with a group name beginning with the generic name, and matching the INSLIB parameter value, are shown. If an asterisk is not included with the generic name, the system assumes the value to be the complete group name.

Location of output (OUTPUT)

Enter the location of the output from the DSPCICSFCT command.

Possible values are:

*****: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT:** The output is printed with the job spool output.

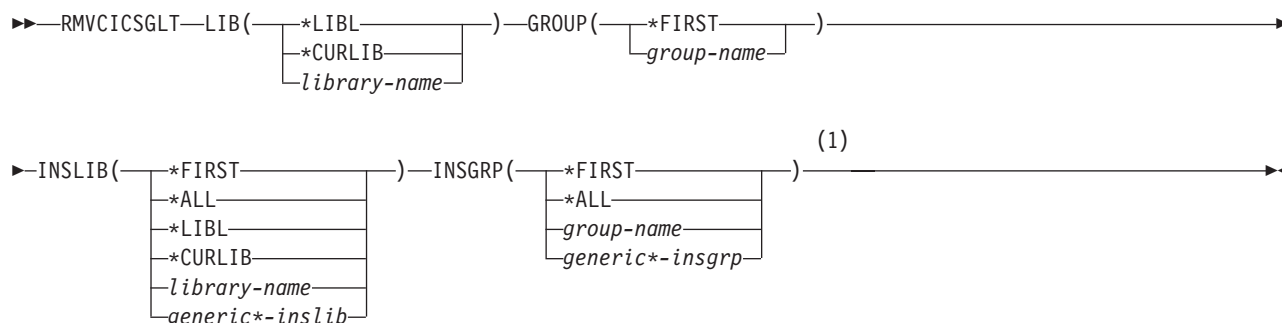
Examples

```
DSPCICSGLT LIB(SAMPLE1) GROUP(ACCT)
```

This command displays the first GLT entry located in group ACCT in OS/400 library SAMPLE1.

Using the RMVCICSGLT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Group List Table (RMVCICSGLT) command to delete an entry from the GLT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

- *LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.
- *CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.
- library-name:* Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the GLT entry is to be removed.

Possible values are:

- *FIRST:** No group is specified, the first group found is used.
- group-name:* The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Install library (INSLIB)

Identifies the first part of the GLT entry to be removed. This is also known as the OS/400 library that contains the group to be used when starting the control region.

Possible values are:

- *FIRST:** Remove the first GLT entry that matches the INSGRP parameter value.
- *ALL:** Remove all of the GLT entries that match the INSGRP parameter value.
- *LIBL:** Remove all of the GLT entries that have *LIBL specified and that match the INSGRP parameter value.

***CURLIB:** Remove all of the GLT entries that have *CURLIB specified and that match the INSGRP parameter value.

library-name: The library name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-inslib:* Specify the generic name of the OS/400 library. A generic name is a string of one or more characters followed by an (*); for example, ABC*. If a generic name is specified, then all GLT entries with an OS/400 library name beginning with the generic name, and matching the INSGRP parameter value, are removed. If an asterisk is not included with the generic name, the system assumes the value to be the complete OS/400 library name.

Install group (INSGRP)

Identifies the second part of the GLT entry to be removed. This is also known as the group to be used when starting the control region.

Possible values are:

***FIRST:** Remove the first GLT entry that matches the INSLIB parameter value.

***ALL:** Remove all of the GLT entries that match the INSLIB parameter value.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-insgrp:* Specify the generic name of the group. A generic name is a string of one or more characters followed by an asterisk(*); for example, ABC*. If a generic name is specified, then all GLT entries with a group name beginning with the generic name, and matching the INSLIB parameter value, are removed. If an asterisk is not included with the generic name, the system assumes the value to be the complete group name.

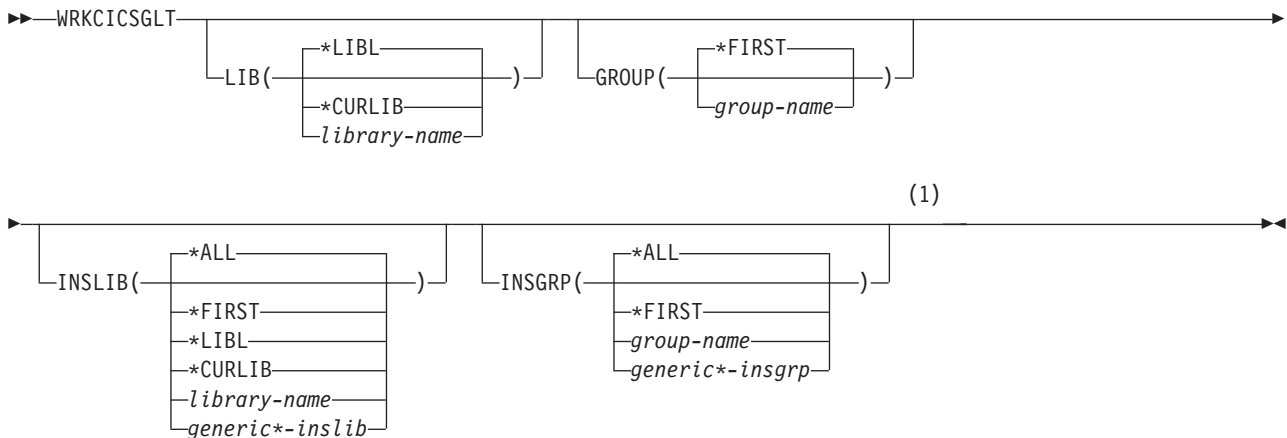
Examples

```
RMVCICSGLT LIB(SAMPLE1) GROUP(ACCT)
           INSLIB(QCICS) INSGRP(AEGEDF)
```

This command removes the GLT entry called QCICS AEGEDF, located in group ACCT in OS/400 library SAMPLE1.

Using the WRKCICSGLT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Group List Table (WRKCICSGLT) command to list entries in the GLT. You can remove copy or display entries, or add new entries to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the GLT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Install library (INSLIB)

Enter the name of the first part of the GLT entry to be listed. This could also be known as the OS/400 library that contains the group to be used when starting the CICS/400 control region.

Possible values are:

***ALL:** List all the GLT entries that match the INSGRP parameter value.

***FIRST:** List the first GLT entry that matches the INSGRP parameter value.

***LIBL:** List all GLT entries that have *LIBL specified and that match the INSGRP parameter value.

***CURLIB:** List all GLT entries that have *CURLIB specified and that match the INSGRP parameter value.

library-name: The library name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-inslib:* Specify the generic name of the OS/400 library. A generic name is a string of one or more characters followed by an (*); for example, ABC*. If a generic name is specified, then all GLT entries with the OS/400 library name beginning with the generic name, and matching the INSGRP parameter value, are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete OS/400 library name.

Install group (INSGRP)

Enter the name of the second part of the GLT entry to be listed. This could also be known as the group to be used when starting the CICS/400 control region.

Possible values are:

***ALL:** List all GLT entries that match the INSLIB parameter value.

***FIRST:** List the first GLT entry that matches the INSLIB parameter value.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-insgrp:* Specify the generic name of the group. A generic name is a string of one or more characters followed by an asterisk(*); for example, ABC*. If a generic name is specified, then all GLT entries with the group name beginning with the generic name, and matching the INSLIB parameter value, are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete group name.

Examples

```
WRKCICSGLT LIB(SAMPLE1) GROUP(ACCT)
```

This command lists all GLT entries located in group ACCT in OS/400 library SAMPLE1.

Managing journal resource definitions

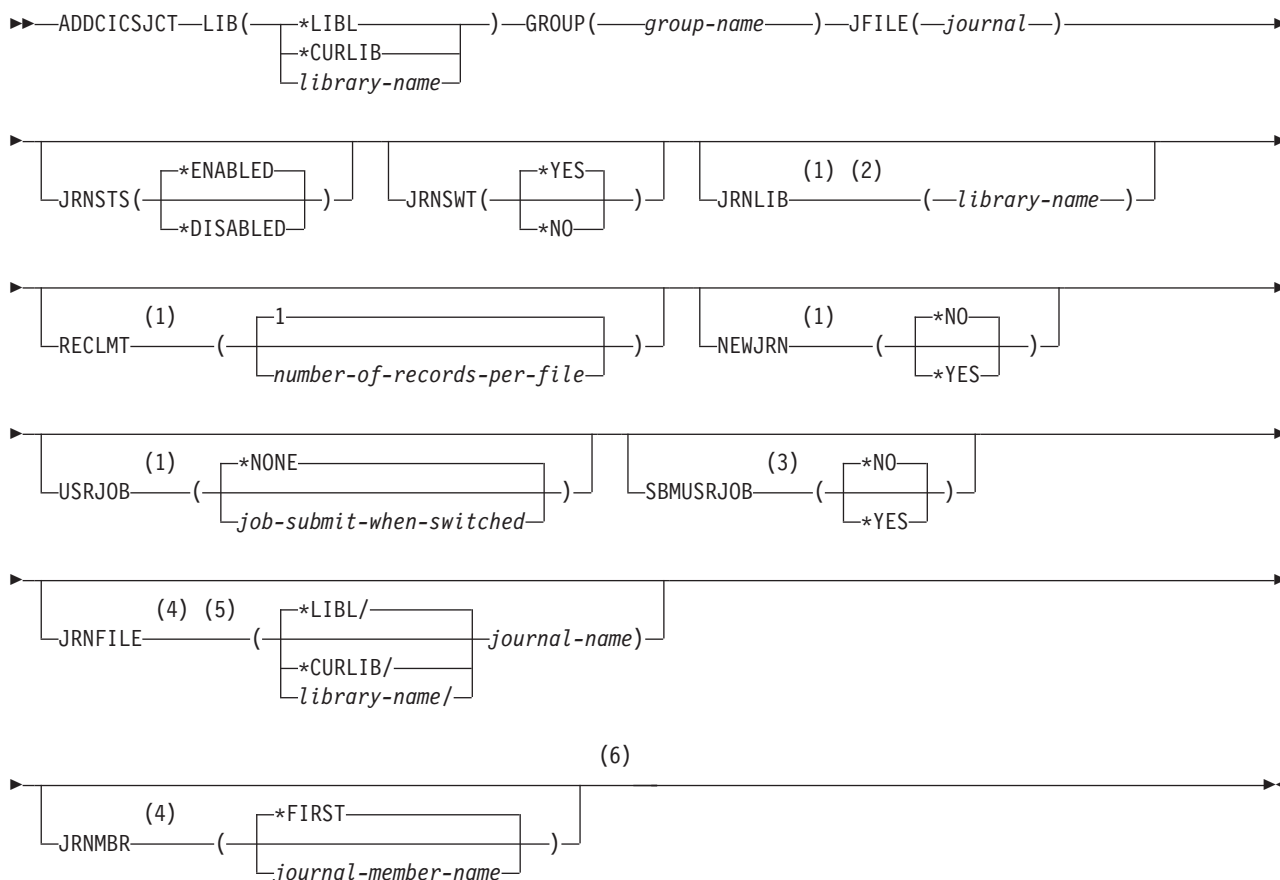
When a file is defined in the FCT, you can specify whether or not activity on that file is to be journaled and the ID of the journal file. Journal files are identified by a number in the range 1 through 99. In the journal control table (JCT), you define the characteristics of each journal file.

See “Journal control considerations” on page 96 for guidance information.

ADDCICSJCT

Using the ADDCICSJCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The JRNLIB parameter, RECLMT parameter, NEWJRN parameter, and USRJOB parameter are valid only when JRNSWT(*YES) is specified.
- 2 The JRNLIB parameter is required when JRNSWT(*YES) is specified.
- 3 The SBMUSRJOB parameter is valid only when USRJOB(*NONE) is not specified.
- 4 The JRNFILE parameter and JRNMBR parameter are valid only when JRNSWT(*NO) is specified.
- 5 The JRNFILE parameter is required when JRNSWT(*NO) is specified.
- 6 All parameters preceding this point can be specified positionally.

Function

Use the Add CICS/400 Journal Control Table (ADDCICSJCT) command to add an entry to the JCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this JCT entry is to be added.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Number (JFILE)

Enter the user journal number. This user journal may also be used for automatic journaling of file activity. The number is also used to identify this JCT entry.

journal: A number in the range 1 through 99.

Optional parameters

Status (JRNSTS)

Indicates whether or not the journal number can be used.

Possible values are:

***ENABLED:** The journal number can be used.

***DISABLED:** The journal number cannot be used.

Automatic switching (JRNSWT)

Indicates whether or not the journal can be switched automatically, when it is full, to the next file generation

Possible values are:

***YES:** The journal will be switched automatically when it is full.

***NO:** The journal will not be switched automatically when it is full.

Library (JRNLIB)

The OS/400 library name that will contain the journal. This parameter is valid only when the JRNSWT parameter contains *YES.

Note: The OS/400 file name that is used will be generated by the CICS/400 control region. The OS/400 file name is composed of the mask: AEGJ**C***nm***xxx**, where *nm* is the journal number and *xxx* is the generation number.

library-name: The library name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Record capacity (RECLMT)

The number of records accumulated in order to switch to the next generation of the journal. This parameter is valid only when the JRNSWT parameter contains *YES.

Possible values are:

1: The journal is switched when one record is accumulated.

number-of-records-per-file: A number in the range 1 through 99 999.

Switch journal at startup (NEWJRN)

Indicates whether or not the journal will be switched to the next generation

ADDCICSJCT

when the CICS/400 control region is started. This parameter is valid only when the JRNSWT parameter contains *YES.

Possible values are:

***NO:** Switch only when the journal is full.

***YES:** Switch when the CICS/400 control region is started normally and when the journal is full.

Submit user job when switched (USRJOB)

The OS/400 program that is submitted via the SBMUSRJOB CL command when the journal is switched. This parameter is valid only when the JRNSWT parameter contains *YES.

Note: This OS/400 submitted program will not accept any parameters and the program object must exist in the library list associated with the CICS/400 control region.

Possible values are:

***NONE:** No OS/400 job is submitted when the journal is switched.

job-submit-when-switched: The maximum length is 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Submit user job at shutdown (SBMUSRJOB)

Indicates whether the OS/400 job is submitted when the CICS/400 control region ends. This is not valid when the USRJOB parameter contains *NONE.

Possible values are:

***NO:** Do not submit the OS/400 job when CICS/400 control region ends.

***YES:** Submit the OS/400 job when CICS/400 control region ends.

File (JRNFILE)

The name of the file that will be utilized by the journal number. This parameter is valid only when the JRNSWT parameter contains *NO.

Possible library values are:

***LIBL:** The library list for the job that is associated to the CICS/400 control region is used to locate the file.

***CURLIB:** The current library for the job that is associated to the CICS/400 control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

journal-name: Specify the name of the file.

Member (JRNMBR)

The name of the member in the file that will be utilized by the journal number. This parameter is valid only when the JRNSWT parameter contains *NO.

Possible values are:

***FIRST:** No file member is specified. The first member in the file is used.

journal-member-name: Specify the name of the file member.

Examples

```
ADDCICSJCT LIB(CICSWORK) GROUP(ACCT) JFILE(50)
           JRNSTS(*DISABLED) JRNSWT(*NO)
           JRNFILE(CICSWORK/JRNLFIL)
```

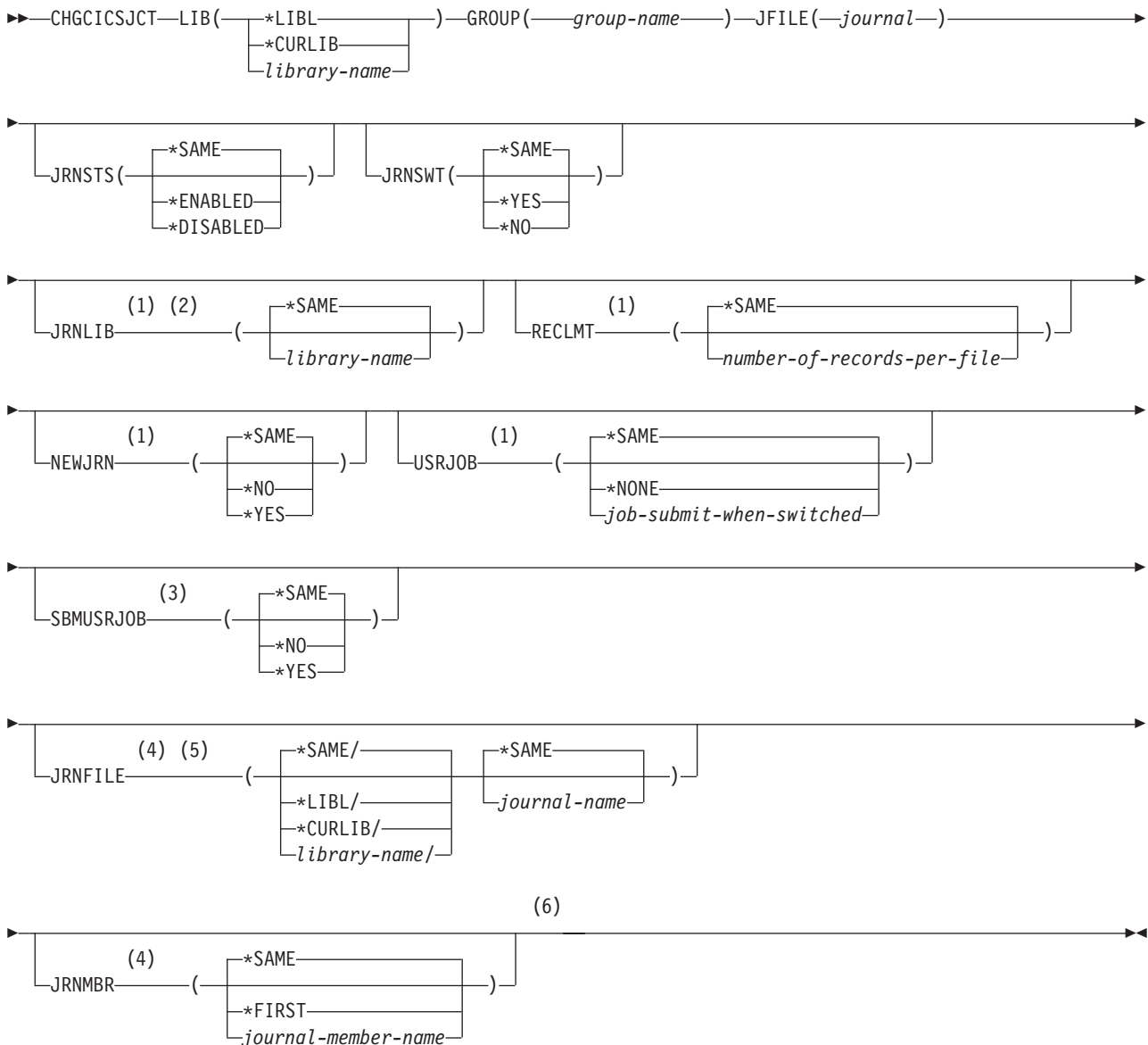
ADDCICSJCT

This command adds a JCT entry called 50 to group ACCT in OS/400 library CICSWORK. The journal file status at control region startup is DISABLED. The journal will not be switched when full and, by default, when the control region is started. The name of journal file 50 is JRNLFIL.

CHGCICSJCT

Using the CHGCICSJCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The JRNLIB parameter, RECLMT parameter, NEWJRN parameter, and USRJOB parameter are valid only when JRNSWT(*YES) is specified.
- 2 The JRNLIB parameter is required when JRNSWT(*YES) is specified.
- 3 The SBMUSRJOB parameter is valid only when USRJOB(*NONE) is not specified.
- 4 The JRNFILE parameter and JRNMBR parameter are valid only when JRNSWT(*NO) is specified.
- 5 The JRNFILE parameter is required when JRNSWT(*NO) is specified.
- 6 All parameters preceding this point can be specified positionally.

Function

Use the Change CICS/400 Journal Control Table (CHGCICSJCT) command to change a JCT entry.

Required parameters**Library (LIB)**

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the JCT entry to be changed.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Number (JFILE)

Enter the user journal number. This user journal may also be used for automatic journaling of file activity. The number is also used to identify this JCT entry.

journal: A number in the range 1 through 99.

Optional parameters**Status (JRNSTS)**

Indicates whether or not the journal number can be used.

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

***ENABLED:** The journal number can be used.

***DISABLED:** The journal number cannot be used.

Automatic switching (JRNSWT)

Indicates whether or not the journal can be switched automatically, when it is full, to the next file generation

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

***YES:** The journal will be switched automatically when it is full.

***NO:** The journal will not be switched automatically when it is full.

Library (JRNLIB)

The OS/400 library name that will contain the journal. This parameter is valid only when the JRNSWT parameter contains *YES.

Note: The OS/400 file name that is used will be generated by the CICS/400 control region. The OS/400 file name is composed of the mask: AEGJ**Cnnxxx**, where *nn* is the journal number and *xxx* is the generation number.

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

library-name: The library name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Record capacity (RECLMT)

The number of records accumulated in order to switch to the next generation of the journal. This parameter is valid only when the JRNSWT parameter contains *YES.

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

number-of-records-per-file: A number in the range 1 through 99 999.

Switch journal at startup (NEWJRN)

Indicates whether or not the journal will be switched to the next generation when the CICS/400 control region is started. This parameter is valid only when the JRNSWT parameter contains *YES.

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

***NO:** Switch only when the journal is full.

***YES:** Switch when the CICS/400 control region is started normally and when the journal is full.

Submit user job when switched (USRJOB)

The OS/400 program that is submitted via the SBMUSRJOB CL command when the journal is switched. This parameter is valid only when the JRNSWT parameter contains *YES.

Note: This OS/400 submitted program will not accept any parameters and the program object must exist in the library list associated with the CICS/400 control region.

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

***NONE:** No OS/400 job is submitted when the journal is switched.

job-submit-when-switched: The maximum length is 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Submit user job at shutdown (SBMUSRJOB)

Indicates whether or not the OS/400 job will be submitted when the CICS/400 control region ends. This is not valid when the USRJOB parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

***NO:** Do not submit the OS/400 job when CICS/400 control region ends.

***YES:** Submit the OS/400 job when CICS/400 control region ends.

File (JRNFIL)

The name of the file that will be utilized by the journal number. This parameter is valid only when the JRNSWT parameter contains *NO.

Possible library values are:

***SAME:** Keep the value currently specified in the JCT entry.

***LIBL:** The library list for the job that is associated to the CICS/400 control region is used to locate the file.

***CURLIB:** The current library for the job that is associated to the CICS/400 control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

Possible file name values are:

***SAME:** Keep the value currently specified in the JCT entry.

journal-name: Specify the name of the file.

Member (JRNMBR)

The name of the member in the file that will be utilized by the journal number. This parameter is valid only when the JRNSWT parameter contains *NO.

Possible values are:

***SAME:** Keep the value currently specified in the JCT entry.

***FIRST:** No file member is specified. The first member in the file is used.

journal-member-name: Specify the name of the file member.

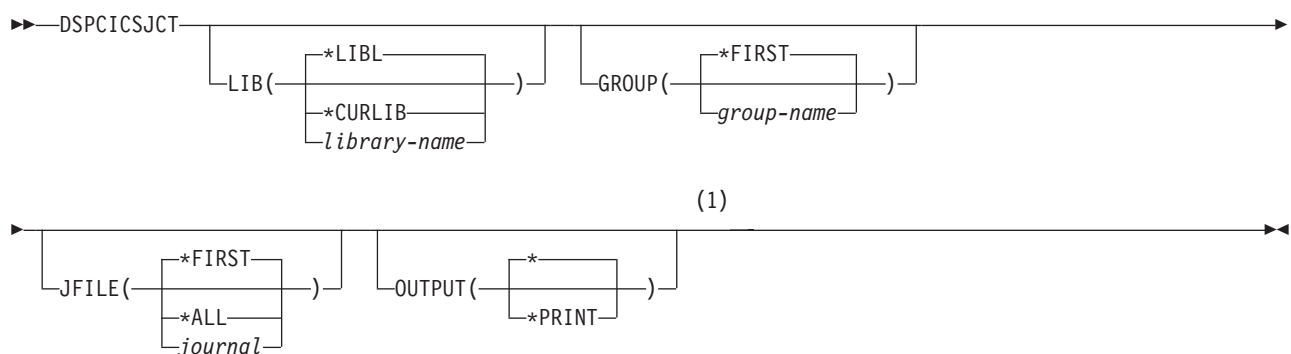
Examples

```
CHGCICSJCT LIB(CICSWORK) GROUP(ACCT)
           JFILE(50) JRNSTS(*ENABLED) JRNSWT(*YES) RECLMT(50000)
```

This command changes the JCT entry called 50 located in group ACCT in OS/400 library CICSWORK. Journal file 50 is now enabled. The journal will switch to the next generation when the record limit of 50 000 is reached or the journal file is full, and on control region startup.

Using the DSPCICSJCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Journal Control Table (DSPCICSJCT) command to display a JCT entry. You can only view this entry; you can neither make changes to it nor delete it.

DSPCICSJCT

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the JCT entry to be displayed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Number (JFILE)

The name of the JCT entry to be displayed.

Possible values are:

***FIRST:** Display the first JCT entry.

***ALL:** Display all the JCT entries.

journal: A number in the range 1 through 99.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*****: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT:** The output is printed with the job spool output.

Examples

```
DSPCICSJCT LIB(CICSWORK) GROUP(ACCT) JFILE(*ALL)
```

This command displays all JCT entries located in group ACCT in OS/400 library CICSWORK.

Using the RMVCICSJCT command

Job: B,I Pgm: B,I REXX: B,I Exec

```
▶▶ RMVCICSJCT LIB( [ *LIB ] ) GROUP( [ *FIRST ] ) JFILE( [ *FIRST ] ) (1)
                   [ *CURLIB ]
                   [ library-name ]
                   [ group-name ]
                   [ *ALL ]
                   [ journal ]
```

Notes:

1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Journal Control Table (RMVCICSJCT) command to delete an entry from the JCT.

Required parameters**Library (LIB)**

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the JCT entry to be removed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Number (JFILE)

The name of the JCT entry to be removed.

Possible values are:

***FIRST:** Remove the first JCT entry.

***ALL:** Remove all JCT entries.

journal: A number in the range 1 through 99.

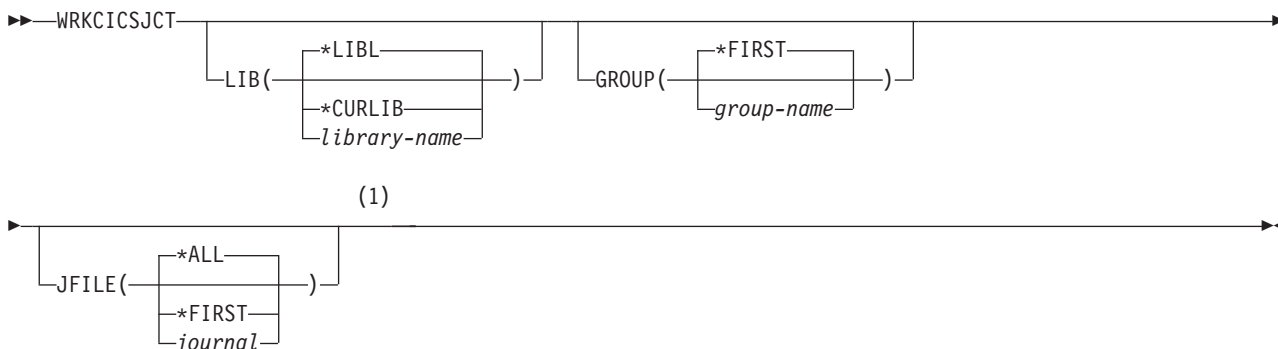
Examples

```
RMVCICSJCT LIB(CICSWORK) GROUP(ACCT)
JFILE(02)
```

This command removes the JCT entry called 02 from group ACCT in OS/400 library CICSWORK.

Using the WRKCICSJCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Journal Control Table (WRKCICSJCT) command to list entries in the JCT. You can change, remove, copy or display entries in the list, or add new entries to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the JCT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Number (JFILE)

Enter the name of the JCT entry to be listed. This is also the journal number used for automatic journaling of files.

Possible values are:

***ALL:** List all JCT entries.

***FIRST:** List the first JCT entry.

journal: Enter a number in the range 1 through 99.

Examples

```
WRKCICSJCT LIB(CICSWORK) GROUP(ACCT)
```

This command lists all JCT entries located in group ACCT in OS/400 library CICSWORK.

Managing transaction definitions

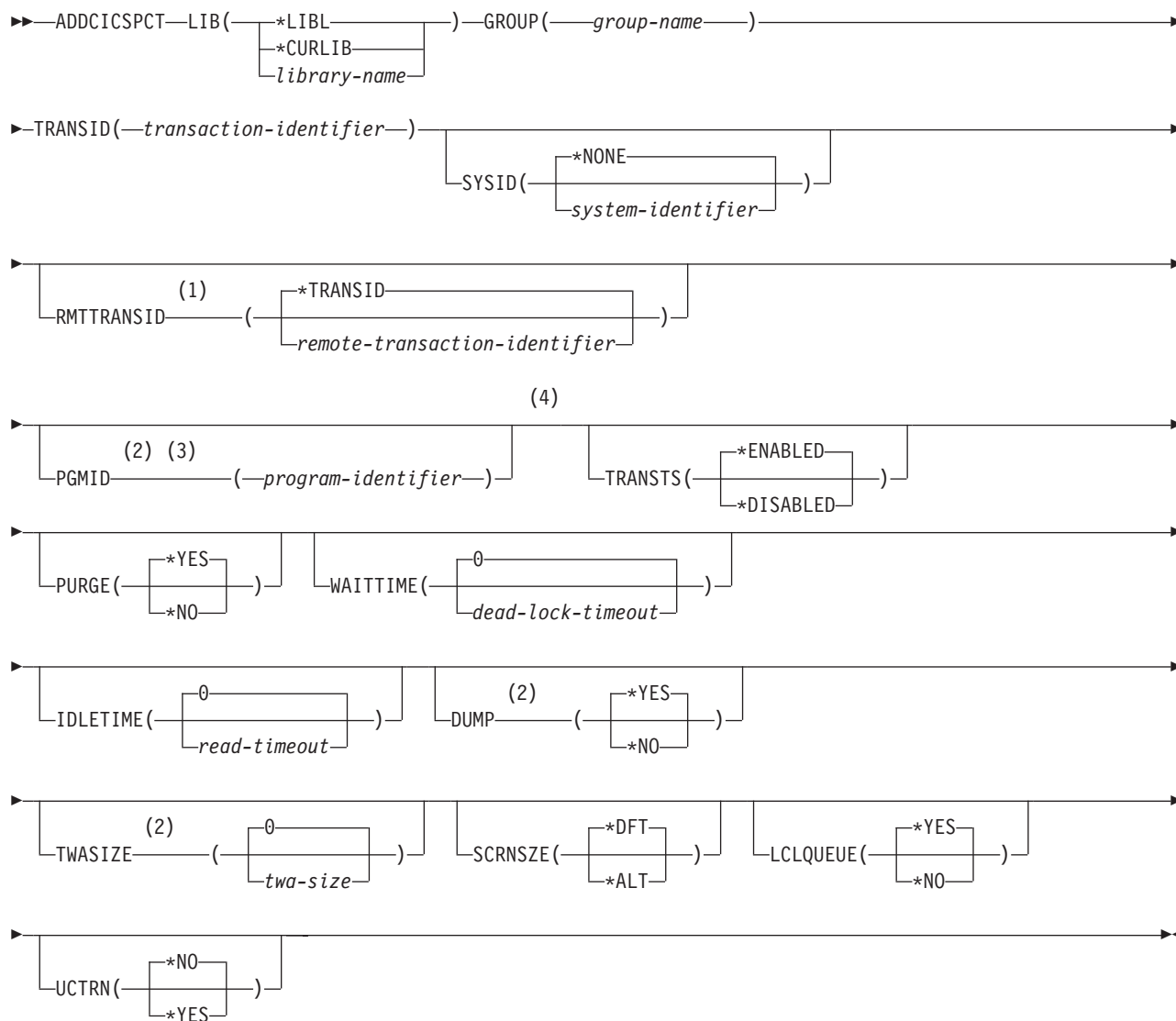
The program control table (PCT) contains a definition for each transaction that may be run from this system. Each local transaction must be linked with a program that is defined in the processing program table (PPT). Each remote transaction should have an associated remote system identifier that is defined in the terminal control table (System Entry) (TCS).

See “Transaction, program, and map considerations” on page 82 for guidance information.

ADDCICSPCT

Using the ADDCICSPCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The RMTTRANSID parameter is not valid when SYSID(*NONE) is specified.
- 2 The PGMID parameter, DUMP parameter, and TWASIZE parameter are valid only when SYSID(*NONE) is specified.
- 3 The PGMID parameter is required when SYSID(*NONE) is specified.
- 4 All parameters preceding this point can be specified positionally.

Function

Use the Add CICS/400 Program Control Table (ADDCICSPCT) command to add an entry to the PCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this PCT entry is to be added.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Transaction (TRANSID)

Enter the transaction identifier used to start a program defined in the processing program table. This name is also used to identify this PCT entry.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

Optional parameters

CICS system (SYSID)

For a remote transaction, enter the identifier of the system that owns the transaction. The system should have a TCS entry. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***NONE:** The transaction is held on the local system.

system-identifier: The remote system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote transaction (RMTTRANSID)

Enter the identifier by which the transaction is known on the remote system. This parameter is not valid for a local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***TRANSID:** The local and remote transaction identifiers are the same. The contents of the **Transaction (TRANSID)** parameter in this PCT entry will be used.

remote-transaction-identifier: The remote transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after

the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

CICS program (PGMID)

Enter the identifier of the program to be started when the transaction is used. The program should be defined in the PPT. This parameter is only valid for local transactions, that is when the **CICS system (SYSID)** parameter contains *NONE.

program-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Status (TRANSTS)

Indicates whether or not the transaction can be used.

Possible values are:

***ENABLED:** The transaction can be used.

***DISABLED:** The transaction cannot be used.

Can be purged while executing (PURGE)

Indicates whether or not the transaction may be purged.

Possible values are:

***YES:** The transaction may be purged.

***NO:** The transaction may not be purged.

Maximum deadlock wait time (WAITTIME)

Enter the amount of time in minutes and seconds that the transaction can be in a dead lock state. The format is *mmss*.

Possible values are:

0: The transaction can be in a dead lock state for an unlimited amount of time.

dead-lock-timeout: Enter a number in the range 0 through 7000 (70 minutes).

Maximum device I/O wait time (IDLETIME)

Enter the amount of time in minutes and seconds that the transaction may wait for terminal input or output. The format is *mmss*.

Possible values are:

0: The transaction may wait for unlimited amount of time for terminal input or output.

read-timeout: A number in the range 0 through 7000 (70 minutes).

Dump when abend (DUMP)

Indicates whether or not a transaction dump is taken when the transaction terminates abnormally. This parameter is valid only for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***YES:** A transaction dump will be taken on abnormal termination of the transaction.

***NO:** A transaction dump will not be taken on abnormal termination on the transaction.

Transaction work area size (TWASIZE)

Enter the size of the associated transaction work area (TWA). This parameter is valid only for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

0: There is no TWA associated with the transaction.

twa-size: A number in the range 0 through 32 767.

Screen size used (SCRNSZE)

Indicates whether the default or the alternate screen size is to be used for the terminal running this transaction.

Possible values are:

*DFT: The default screen size is to be used.

*ALT: The alternate screen size is to be used.

Local system queuing (LCLQUEUE)

Indicates whether or not queuing on the local system is to be performed.

Possible values are:

*YES: Local system queuing is to be performed.

*NO: Local system queuing is not to be performed.

Auto uppercase translation (UCTRN)

Indicates whether or not lowercase terminal input is to be translated to uppercase before being passed to the application program.

Possible values are:

*NO: Lowercase terminal input for this transaction is not translated to uppercase unless the terminal requires uppercase translation (UCTRN option in the TCT).

*YES: Lowercase terminal input for this transaction is always translated to uppercase.

For the interaction between the UCTRN options of the TCT and PCT, see Table 24 on page 274.

Examples

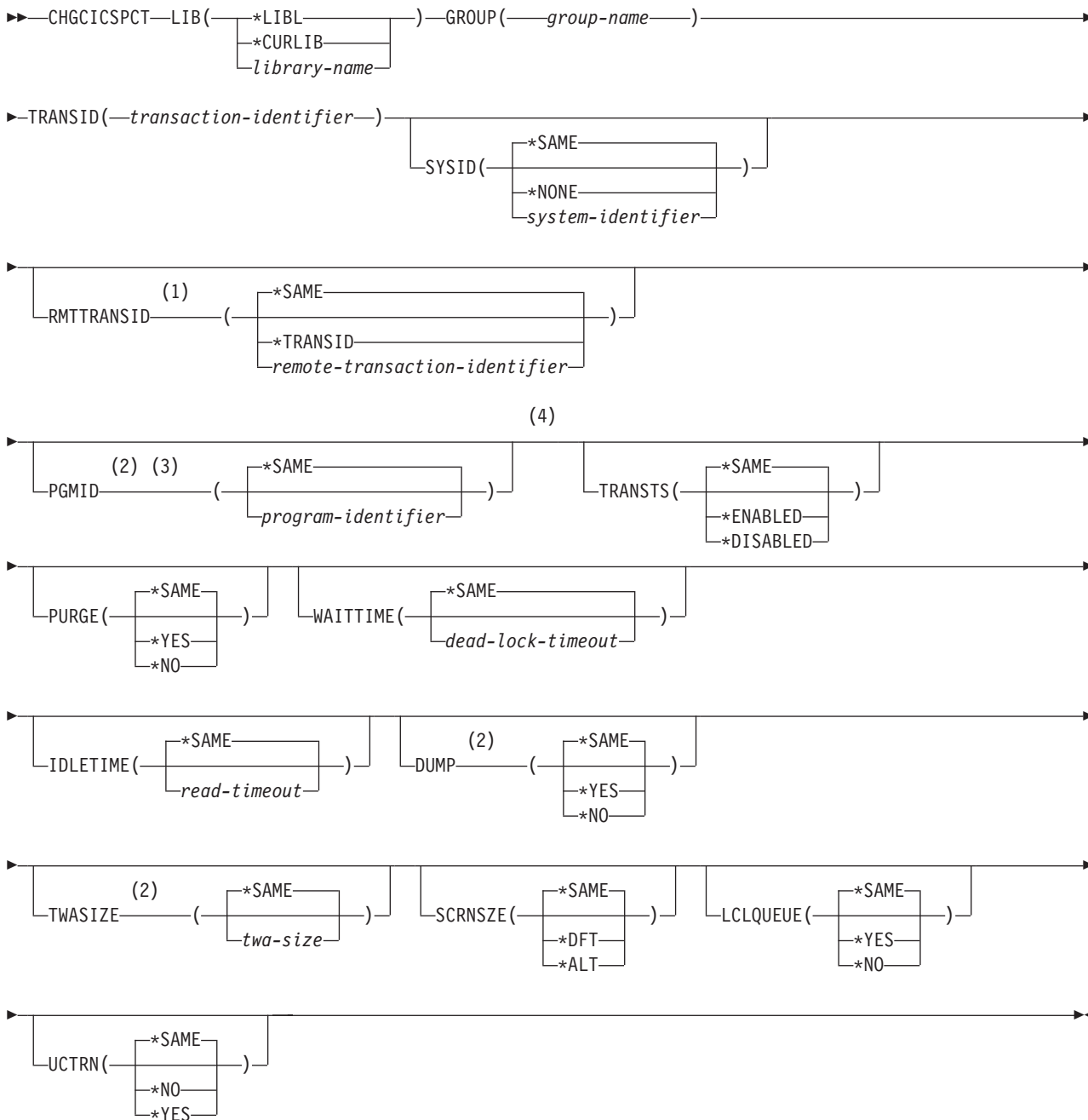
```
ADDCICSPCT LIB(CICSWORK) GROUP(ACCT)
  TRANSID(ACCT) TRANSTS(*DISABLED) PURGE(*NO)
  PGMID(ACCT00)
  SCRNSZE(*ALT) LCLQUEUE(*NO)
```

This command adds a PCT entry called ACCT to group ACCT in OS/400 library CICSWORK. The program identifier (PGMID) ACCT00 will be used to identify the OS/400 program object that is executed when the transaction is used. The program identifier (PGMID) is defined in the PPT.

CHGCICSPCT

Using the CHGCICSPCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The RMTTRANSID parameter is not valid when SYSID(*NONE) is specified.
- 2 The PGMID parameter, DUMP parameter, and TWASIZE parameter are valid only when SYSID(*NONE) is specified.
- 3 The PGMID parameter is required when SYSID(*NONE) is specified.
- 4 All parameters preceding this point can be specified positionally.

Function

Use the Change CICS/400 Program Control Table (CHGCICSPCT) command to change an entry in the PCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PCT entry to be changed.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Transaction (TRANSID)

Enter the transaction identifier of the PCT entry to be changed.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored.

Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

Optional parameters

CICS system (SYSID)

Enter the name of the remote system that owns the transaction. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***SAME:** Keep the value currently specified in the PCT entry.

***NONE:** The transaction is held on the local system.

system-identifier: The remote system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote transaction (RMTTRANSID)

Enter the identifier by which the transaction is known on the remote system.

This parameter is not valid for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in the PCT entry.

***TRANSID:** The local and remote transaction identifiers are the same. The name in the **Transaction (TRANSID)** parameter will be used.

remote-transaction-identifier: The remote transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

CICS program (PGMID)

Enter that name of the program to be started by this transaction identifier. This program should have a PPT entry. This parameter is valid only for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in the PCT entry.

program-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Status (TRANSTS)

Indicates whether or not the transaction may be used.

Possible values are:

***SAME:** Keep the value currently specified in the PCT entry.

***ENABLED:** The transaction may be used.

***DISABLED:** The transaction may not be used.

Can be purged while executing (PURGE)

Indicates whether or not the transaction may be purged.

Possible values are:

***SAME:** Keep the value currently specified in the PCT entry.

***YES:** The transaction may be purged.

***NO:** The transaction may not be purged.

Maximum deadlock wait time (WAITTIME)

Enter the amount of time in minutes and seconds that the transaction can be in a dead lock state. The format is *mmss*.

Possible values are:

***SAME:** Keep the value currently specified in the PCT entry.

dead-lock-timeout: A number in the range 0 through 7000 minutes). When the value specified is 0, then the transaction identifier can be in a dead lock state for an unlimited amount of time.

Maximum device I/O wait time (IDLETIME)

Enter the amount of time in minutes and seconds that the transaction waits for terminal input or output. The format is *mmss*.

Possible values are:

***SAME:** Keep the value currently specified in the PCT entry.

read-timeout: A number in the range 0 through 7000 (70 minutes). When the value specified is a 0, then the transaction identifier can wait for unlimited amount of time for terminal input or output.

Dump when abend (DUMP)

Indicates whether or not a transaction dump is taken when the transaction terminates abnormally. This parameter is valid only for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME**: Keep the value currently specified in the PCT entry.

***YES**: Take a transaction dump.

***NO**: Do not take a transaction dump.

Transaction work area size (TWSIZE)

Enter the size of the associated Transaction Work Area (TWA). This parameter is valid only for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME**: Keep the value currently specified in the PCT entry.

twa-size: A number in the range 0 through 32 767. Enter 0 if there is no TWA associated with the transaction.

Screen size used (SCRNSZE)

Indicates whether the default or the alternate screen size associated with the terminal to be used to run the transaction.

Possible values are:

***SAME**: Keep the value currently specified in the PCT entry.

***DFT**: The default screen size is to be used.

***ALT**: The alternate screen size is to be used.

Local system queuing (LCLQUEUE)

Indicates whether or not queuing on the local system is to be performed.

Possible values are:

***SAME**: Keep the value currently specified in the PCT entry.

***YES**: Local system queuing is to be performed.

***NO**: Local system queuing is not to be performed.

Auto uppercase translation (UCTRN)

Indicates whether or not lowercase terminal input is to be translated to uppercase before being passed to the application program.

Possible values are:

***SAME**: Keep the value currently specified in the PCT entry.

***NO**: Lowercase terminal input for this transaction is not translated to uppercase unless the terminal requires uppercase translation (UCTRN option in the TCT).

***YES**: Lowercase terminal input for this transaction is always translated to uppercase.

For the interaction between the UCTRN options of the TCT and PCT, see Table 24 on page 274.

CHGCICSPCT

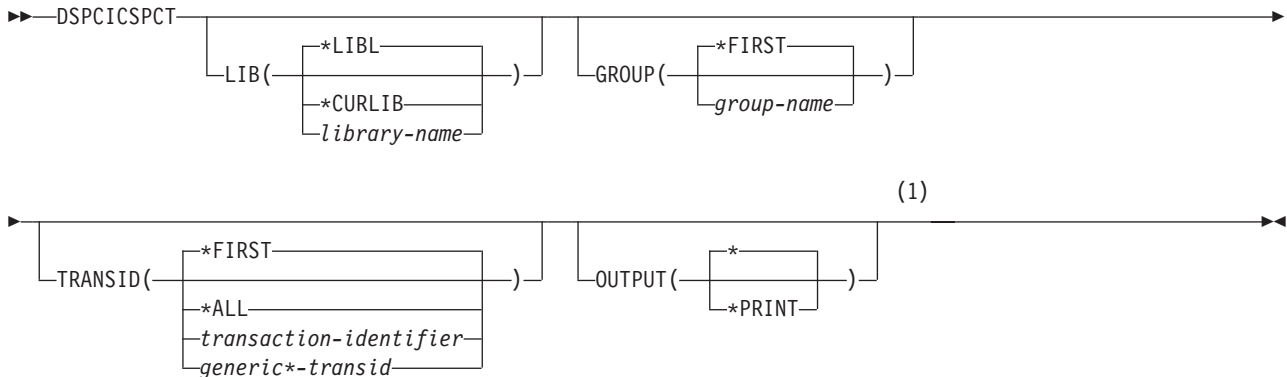
Examples

```
CHGCICSPCT LIB(CICSWORK) GROUP(ACCT)
TRANSID(ACCT) TRANSTS(*ENABLED)
```

This command changes the PCT entry called ACCT located in group ACCT in OS/400 library CICSWORK, to enable transaction identifier ACCT.

Using the DSPCICSPCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Program Control Table (DSPCICSPCT) command to display a PCT entry. You can only view this entry; you can neither change it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PCT entry to be displayed.

Possible values are:

***FIRST:** No CICS/400 group is specified. The first CICS/400 group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Transaction (TRANSID)

Enter the name of the PCT entry to be displayed.

Possible values are:

***FIRST:** Display the first PCT entry.

***ALL:** Display all the PCT entries.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

generic-transid:* Specify the generic name of the CICS/400 transaction identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all PCT entries with transaction identifiers beginning with the generic name are displayed. If an asterisk is not included with the generic name, the system assumes the value to be the complete transaction identifier.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*****: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT:** The output is printed with the job spool output.

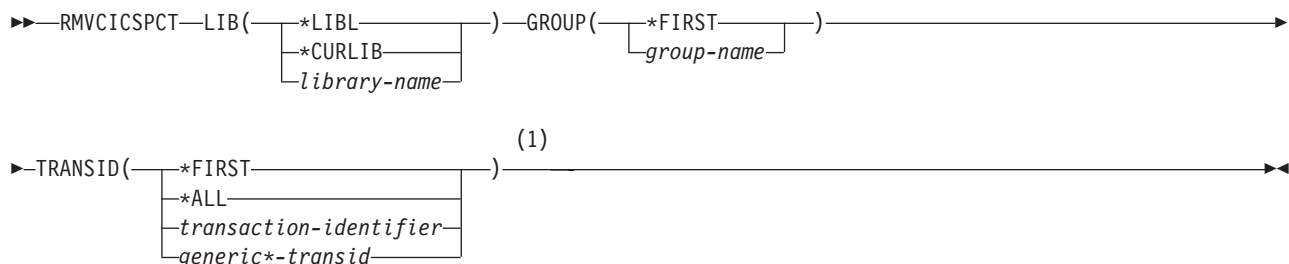
Examples

```
DSPCICSPCT LIB(CICSWORK) GROUP(ACCT) TRANSID(*ALL)
```

This command displays all PCT entries located in group ACCT in OS/400 library CICSWORK.

Using the RMVCICSPCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Program Control Table (RMVCICSPCT) command to delete an entry from the PCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PCT entry to be removed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Transaction (TRANSID)

Enter the name of the PCT entry to be removed.

Possible values are:

***FIRST:** Remove the first PCT entry.

***ALL:** Remove all PCT entries.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

generic-transid:* Specify the generic name of the transaction identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all PCT entries with transaction identifiers beginning the generic name are removed. If an asterisk is not included with the generic (prefix) name, the system assumes the value to be the complete transaction identifier.

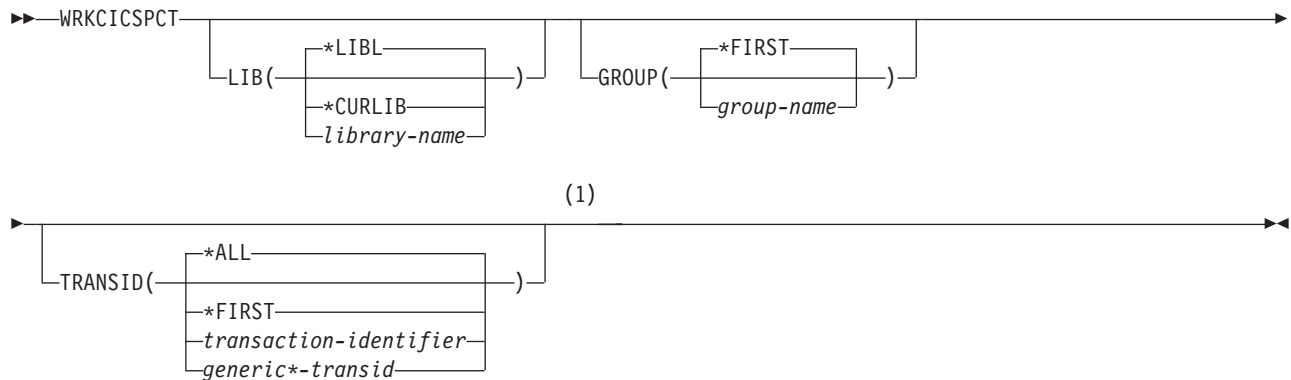
Examples

```
RMVCICSPCT LIB(CICSWORK) GROUP(ACCT) TRANSID(ABC*)
```

This command removes all PPT entries, that start with ABC and end with anything, located in group ACCT in OS/400 library CICSWORK.

Using the WRKCICSPCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Program Control Table (WRKCICSPCT) command to list entries in the PCT. You can change, remove, copy or display entries in the list, or add new ones.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PCT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Transaction (TRANSID)

Enter the name of the PCT entry to be listed. This is the transaction identifier that will be used in EXEC CICS commands to start a CICS/400 program. The program should be defined in the PPT.

Possible values are:

***ALL:** List all PCT entries.

***FIRST:** List the first PCT entry.

WRKCICSPCT

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

generic-transid*: Specify a generic transaction identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all PCT entries with transaction identifiers beginning with the generic name are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete transaction identifier.

Examples

```
WRKCICSPCT LIB(CICSWORK) GROUP(ACCT)
```

This command lists all PCT entries located in group ACCT in OS/400 library CICSWORK.

Managing program definitions

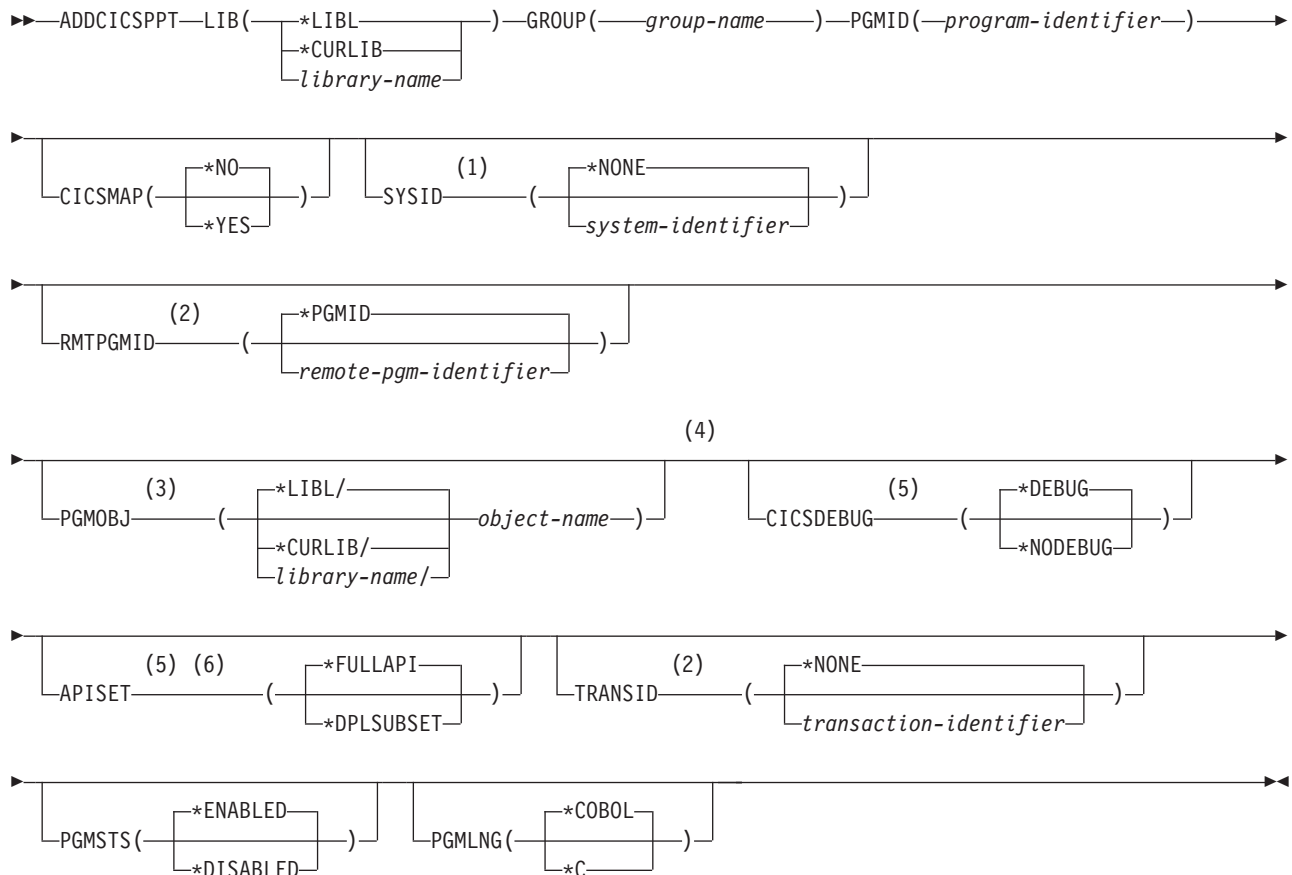
The processing program table (PPT) defines the user program objects that may be invoked by the transaction identifiers held in the PCT. You need to create entries for:

- All programs invoked by a local transaction ID. These programs may be local or remote.
- All programs that are invoked by either the EXEC CICS LINK or the EXEC CICS XCTL commands.
- All BMS map sets used on the local system.

See "Transaction, program, and map considerations" on page 82 for guidance information.

Using the ADDCICSPPT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The SYSID parameter is valid only when CICSMAP(*NO) is specified.
- 2 The RMTPGMID parameter and TRANSID parameter are not valid when SYSID(*NONE) is specified.
- 3 The PGMOBJ parameter is valid, and required, only when SYSID(*NONE) is specified.
- 4 All parameters preceding this point can be specified positionally.
- 5 The CICSDEBUG parameter and APISET parameter are valid only when CICSMAP(*NO) and SYSID(*NONE) are specified.
- 6 The APISET(*DPLSUBSET) is valid only when PGMID does not start with AEG.

Function

Use the Add CICS/400 Processing Program Table (ADDCICSPPT) command to add an entry to the PPT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this PPT entry is to be added.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS program (PGMID)

Enter the program identifier used to initiate an OS/400 program object. This parameter is also used to identify this PPT entry.

program-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

BMS map set (CICSMAP)

Indicates whether or not the program identifier is referencing a BMS map set.

Possible values are:

***NO:** The program identifier is not referencing a BMS map set; it is referencing an application program.

***YES:** The program identifier is referencing a BMS map set.

CICS system (SYSID)

Enter the identifier of the system that owns the program. The system should have a TCS entry. This parameter is valid only for PPT entries defining application programs, that is the CICSMAP parameter should contain *NO. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***NONE:** The program is held on the local system.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote CICS program (RMTPGMID)

Enter the identifier by which the program is known on the remote system. This parameter is not valid for local systems, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***PGMID:** The local and remote program identifier are the same. The local identifier will be used to access the remote program.

remote-pgm-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Program object (PGMOBJ)

Enter the name of the OS/400 program object that will be utilized by the

program identifier. This parameter is valid only for local systems, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible library values are:

***LIBL:** The library list for the job that is associated with the control region is used to locate the OS/400 program object.

***CURLIB:** The current library for the job that is associated with the control region is used to locate the OS/400 program object. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the OS/400 program object is located.

object-name: Specify the name of the OS/400 program object.

CICS debug (CICSDEBUG)

Indicates whether or not the CICS-supplied transaction CEDF can be used when running an OS/400 program object. This parameter is valid only when the PGMID parameter is referencing an application program held on the local system, the CICSMAP parameter contains *NO, and the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***DEBUG:** CEDF can be used with the OS/400 program object.

***NODEBUG:** CEDF cannot be used with the OS/400 program object.

API commands (APISET)

Indicates which of the EXEC CICS commands can be executed in an application program. This parameter is valid only when the CICSMAP parameter refers to an application program held on the local system, that is when the CICSMAP parameter contains *NO and the **CICS system (SYSID)** parameter contains *NONE.

Note: If the program identifier starts with AEG, then this parameter must contain *FULLAPI.

Possible values are:

***FULLAPI:** All the EXEC CICS commands can be used in application programs.

***DPLSUBSET:** Only the EXEC CICS commands specified in the distributed programming link subset can be used in application programs. See the *CICS Family: API Structure* manual for details.

Transaction (TRANSID)

Enter the overriding transaction identifier under which the server program runs during a distributed program link. This transaction identifier should have its own PCT entry. This parameter is not valid for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***NONE:** There is no overriding transaction identifier to be used.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Status (PGMSTS)

Indicates whether or not the program identifier can be used.

ADDCICSPPT

Possible values are:

***ENABLED:** This program identifier can be used.

***DISABLED:** This program identifier cannot be used.

Calling convention (PGMLNG)

Enter the program language convention used to call the OS/400 program object. This determines how CICS/400 passes parameters to the program object.

***COBOL:** CICS/400 uses the CICS COBOL calling convention to invoke the OS/400 program object.

***C** CICS/400 uses the CICS C calling convention to invoke the program object.

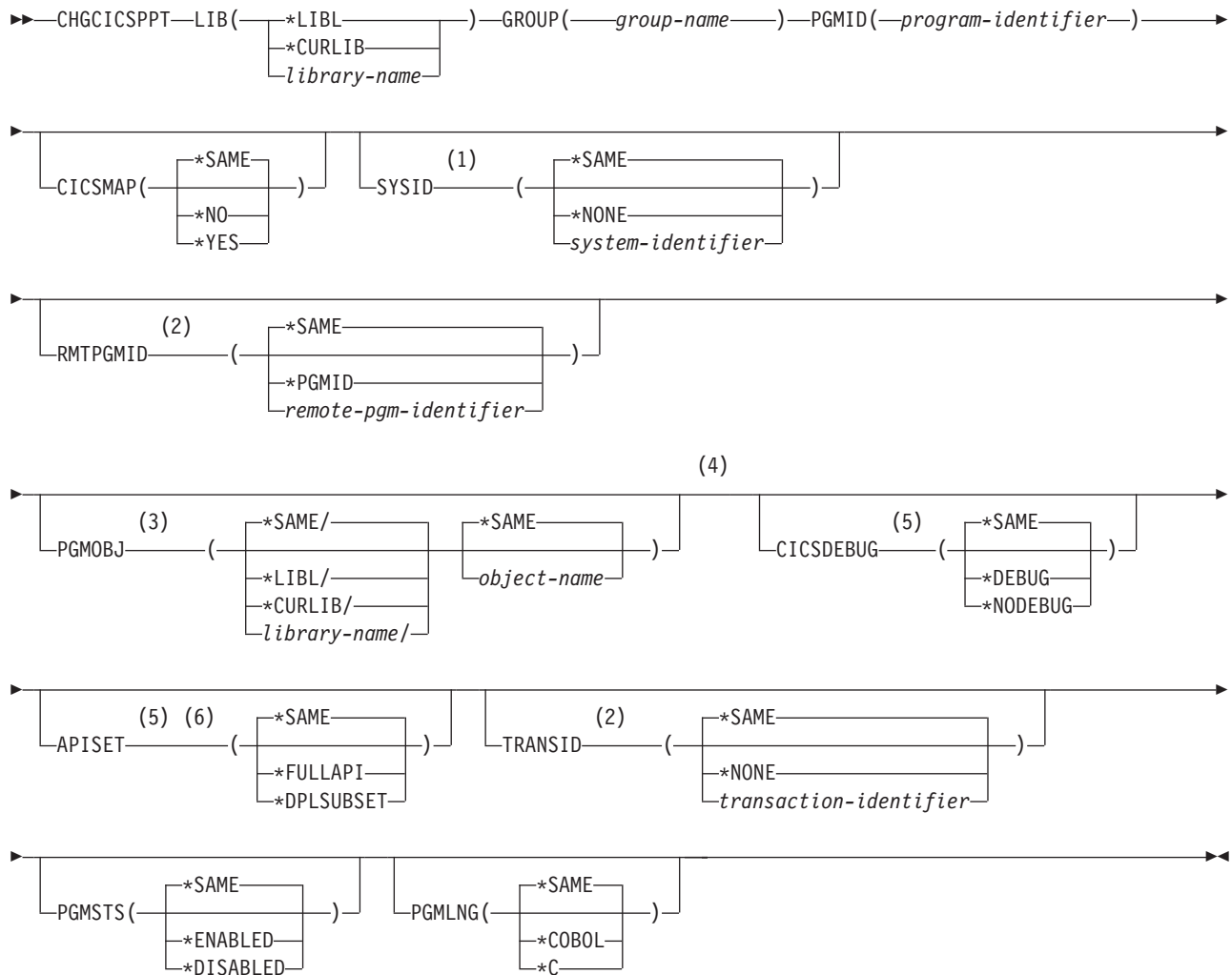
Examples

```
ADDCICSPPT LIB(CICSWORK) GROUP(ACCT)
           PGMID(ACCT01)
           PGMOBJ(OBJLIB/ACCT01)
```

This command adds a PPT entry called ACCT01 to group ACCT in OS/400 library CICSWORK. The OS/400 program object that will be run is ACCT01 in OS/400 library OBJLIB.

Using the CHGCICSPPT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The SYSID parameter is valid only when CICS MAP(*NO) is specified.
- 2 The RMT PG MID parameter and TRANSID parameter are not valid when SYSID(*NONE) is specified.
- 3 The PGM OBJ parameter is valid, and required, only when SYSID(*NONE) is specified.
- 4 All parameters preceding this point can be specified positionally.
- 5 The CICS DEBUG parameter and API SET parameter are valid only when CICS MAP(*NO) and SYSID(*NONE) are specified.
- 6 The API SET(*DPLSUBSET) is valid only when PG MID does not start with AEG.

Function

Use the Change CICS/400 Processing Program Table (CHGCICSPPT) command to change an entry in the PPT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PPT entry to be changed.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS program (PGMID)

Enter the program identifier used to initiate an OS/400 program object. This parameter is also used to identify this PPT entry.

program-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

BMS map set (CICSMAP)

Indicates whether or not the program identifier refers to a BMS map set.

Possible values are:

***SAME:** Keep the value currently specified in the PPT entry.

***NO:** The program identifier is not referencing a BMS map set, that is it is referencing an application program.

***YES:** The program identifier is referencing a BMS map set.

CICS system (SYSID)

Enter the identifier of the remote system owning the program. This parameter is valid only for PPT entries defining application programs, that is when the CICSMAP parameter contains *NO. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***SAME:** Keep the value currently specified in the PPT entry.

***NONE:** The program is held on the local system.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote CICS program (RMTPGMID)

Enter the identifier by which the program is known on the remote system. This parameter is not valid for the local system, that is when the CICS system (SYSID) parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in the PPT entry.

***PGMID:** The local and remote program identifiers are the same. The local identifier will be used to access the program.

remote-pgm-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Program object (PGMOBJ)

Enter the name of the OS/400 program object that will be utilized by the program identifier. This parameter is valid only for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible library values are:

***SAME:** Keep the value currently specified in the PPT entry.

***LIBL:** The library list for the job that is associated with the control region is used to locate the OS/400 program object.

***CURLIB:** The current library for the job that is associated with the control region is used to locate the OS/400 program object. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the OS/400 program object is located.

Possible object values are:

***SAME:** Keep the value currently specified in the PPT entry.

object-name: Specify the name of the OS/400 program object.

CICS debug (CICSDEBUG)

Indicates whether or not the CICS-supplied transaction CEDF can be used when running an OS/400 program object. This parameter is valid only when the PGMID parameter is referencing an application program held on the local system, the CICS MAP parameter contains *NO, and the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Keep the value currently specified in the PPT entry.

***DEBUG:** CEDF can be used with the OS/400 program object.

***NODEBUG:** CEDF cannot be used with the OS/400 program object.

API commands (APISET)

Indicates which of the EXEC CICS commands can be executed in an application program. This parameter is valid only when the CICS MAP parameter references an application program held on the local system, that is when the CICS MAP parameter contains *NO and the **CICS system (SYSID)** parameter contains *NONE.

Note: If the program identifier starts with AEG, then this parameter must contain *FULLAPI.

Possible values are:

***SAME:** Keep the value currently specified in the PPT entry.

***FULLAPI:** All the EXEC CICS commands can be used in application programs.

***DPLSUBSET:** Only the EXEC CICS commands specified in the distributed programming link subset can be used in application programs. See *CICS Family: API Structure* for details.

Transaction (TRANSID)

Enter the overriding transaction identifier under which the server program runs during a distributed program link. This transaction identifier should have its own PCT entry. This parameter is not valid for the local system, that is when the **CICS system (SYSID)** parameter contains *NONE.

Possible values are:

***SAME:** Enter the value currently specified in the PPT entry.

***NONE:** There is no overriding transaction identifier to be used.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. Any characters on the keyboard can be entered, but lowercase letters should be enclosed in apostrophes. Blanks are accepted but, when the entry is defined to the control region, anything after the blank is ignored. Characters may be entered in hexadecimal format, in the form X'A1A2A3A4'. Any value will be accepted but non-printable characters will be rejected when the entry is defined to the control region. See "Entering resource and transaction identifiers" on page 54 for information on how to enter characters in this field.

Status (PGMSTS)

Indicates whether or not the program identifier can be used.

Possible values are:

***SAME:** Keep the value currently specified in the PPT entry.

***ENABLED:** The program identifier can be used.

***DISABLED:** The program identifier cannot be used.

Calling convention (PGMLNG)

Enter the program language convention used to call the OS/400 program object. This determines how CICS/400 passes parameters to the program object.

***SAME:** Keep the value currently specified in the PPT entry.

***COBOL:** CICS/400 uses the CICS COBOL calling convention to invoke the OS/400 program object.

***C** CICS/400 uses the CICS C calling convention to invoke the program object.

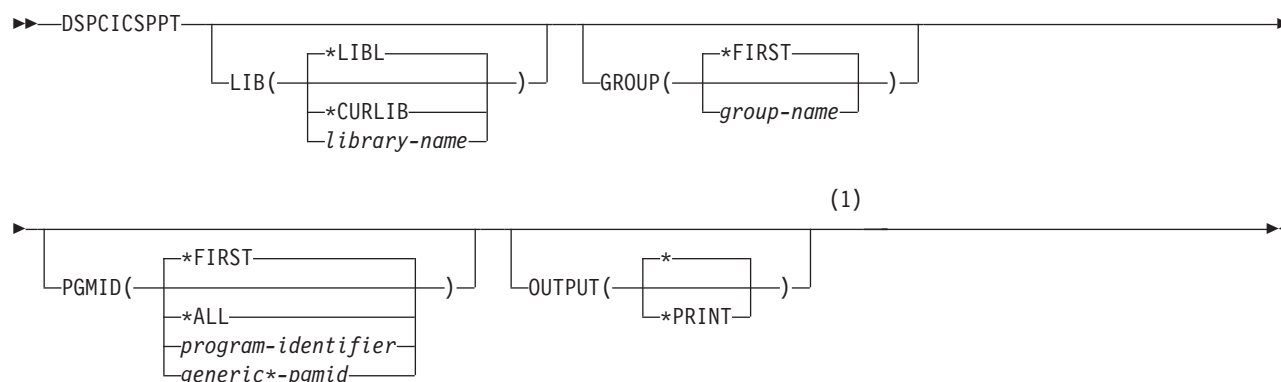
Examples

```
CHGCICSPPT LIB(CICSWORK) GROUP(ACCT)
  PGMID(ACCT01)
  PGMOBJ(*LIBL/ACCT01)
```

This command changes a PPT entry called ACCT01 located in group ACCT in OS/400 library CICSWORK, to use the OS/400 program object ACCT01 in the library list for the job associated with the control region.

Using the DSPCICSPPT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Processing Program Table (DSPCICSPPT) command to display a PPT entry. You can only view this entry; you can neither change it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PPT entry to be displayed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS program (PGMID)

Enter the name of the PPT entry to be displayed.

Possible values are:

***FIRST:** Display the first PPT entry.

***ALL:** Display all the PPT entries.

DSPCICSPPT

program-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-pgmid*: Specify the generic name of a program identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all PPT entries with program identifiers beginning with the generic name are displayed. If an asterisk is not included with the generic name, the system assumes the value to be the complete program identifier.

Location of output (OUTPUT)

Enter the location of the output from the DSPCICSFCT command.

Possible values are:

*: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT**: The output is printed with the job spool output.

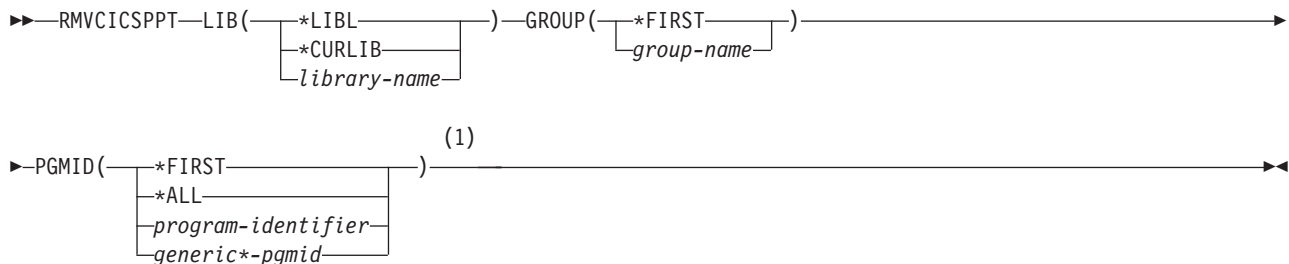
Examples

```
DSPCICSPPT LIB(CICSWORK) GROUP(ACCT)
```

This command displays the first PPT entry located in group ACCT in OS/400 library CICSWORK.

Using the RMVCICSPPT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Processing Program Table (RMVCICSPPT) command to delete an entry from the PPT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL**: The library list is used to locate the first OS/400 library that contains the group.

***CURLIB**: The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PPT entry to be removed.

Possible values are:

***FIRST**: No CICS/400 group is specified. The first CICS/400 group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS program (PGMID)

Enter the identifier of the PPT entry to be removed.

Possible values are:

***FIRST**: Remove the first PPT entry.

***ALL**: Remove all the PPT entries.

program-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-pgmid*: Specify the generic name of the program identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all PPT entries with program identifiers beginning with the generic name are removed. If an asterisk is not included with the generic name, the system assumes the value to be the complete program identifier.

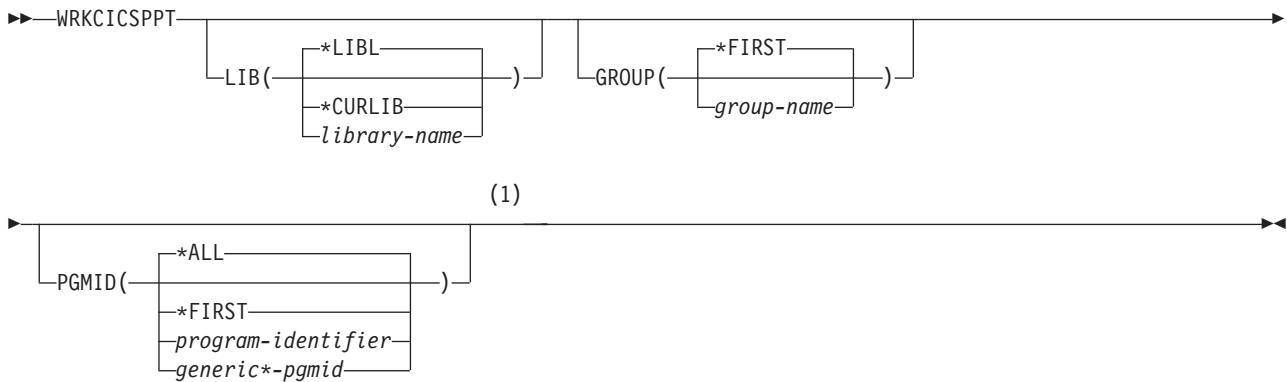
Examples

```
RMVCICSPPT LIB(CICSWORK) GROUP(ACCT)
PGMID(ABC*)
```

This command removes all PPT entries, that start with ABC and end with anything, located in group ACCT in OS/400 library CICSWORK.

Using the WRKCICSPPT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Processing Program Table (WRKCICSPPT) command to list entries in the PPT. You can change, remove, copy, or display entries in the list, or add new ones.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the PPT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS program (PGMID)

Enter the name of the PPT entry to be listed. This is the program identifier used to initiate an OS/400 program object.

Possible values are:

***ALL:** List all PPT entries.

***FIRST:** List the first PPT entry.

program-identifier: The program identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-pgmid*: Specify a generic program identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all PPT entries with program identifiers beginning with the generic name are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete program identifier.

Examples

```
WRKCICSPPT LIB(CICSWORK) GROUP(ACCT)
```

This command lists all entries located in group ACCT in OS/400 library CICSWORK.

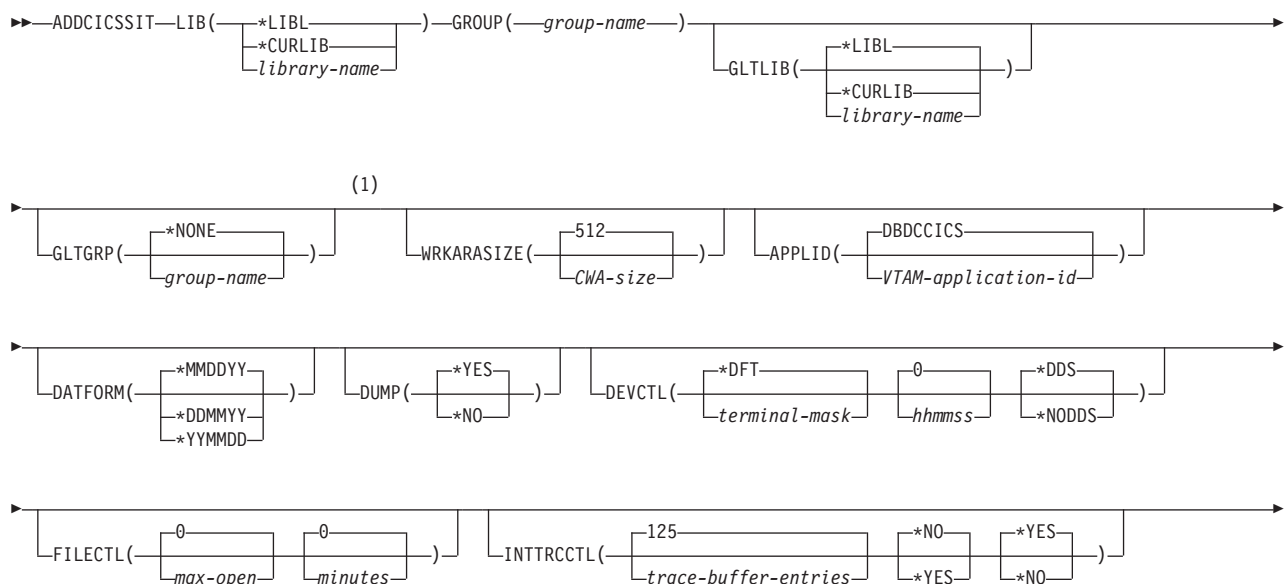
Managing system initialization resource definitions

The system initialization table (SIT) is used to specify the initial startup conditions and control parameters for your system. Because only one set of conditions can apply to the system at any one time, only one SIT entry is allowed per group and per group list table (GLT). You can create as many SIT entries as you like, but each one must belong to a different group and a different GLT.

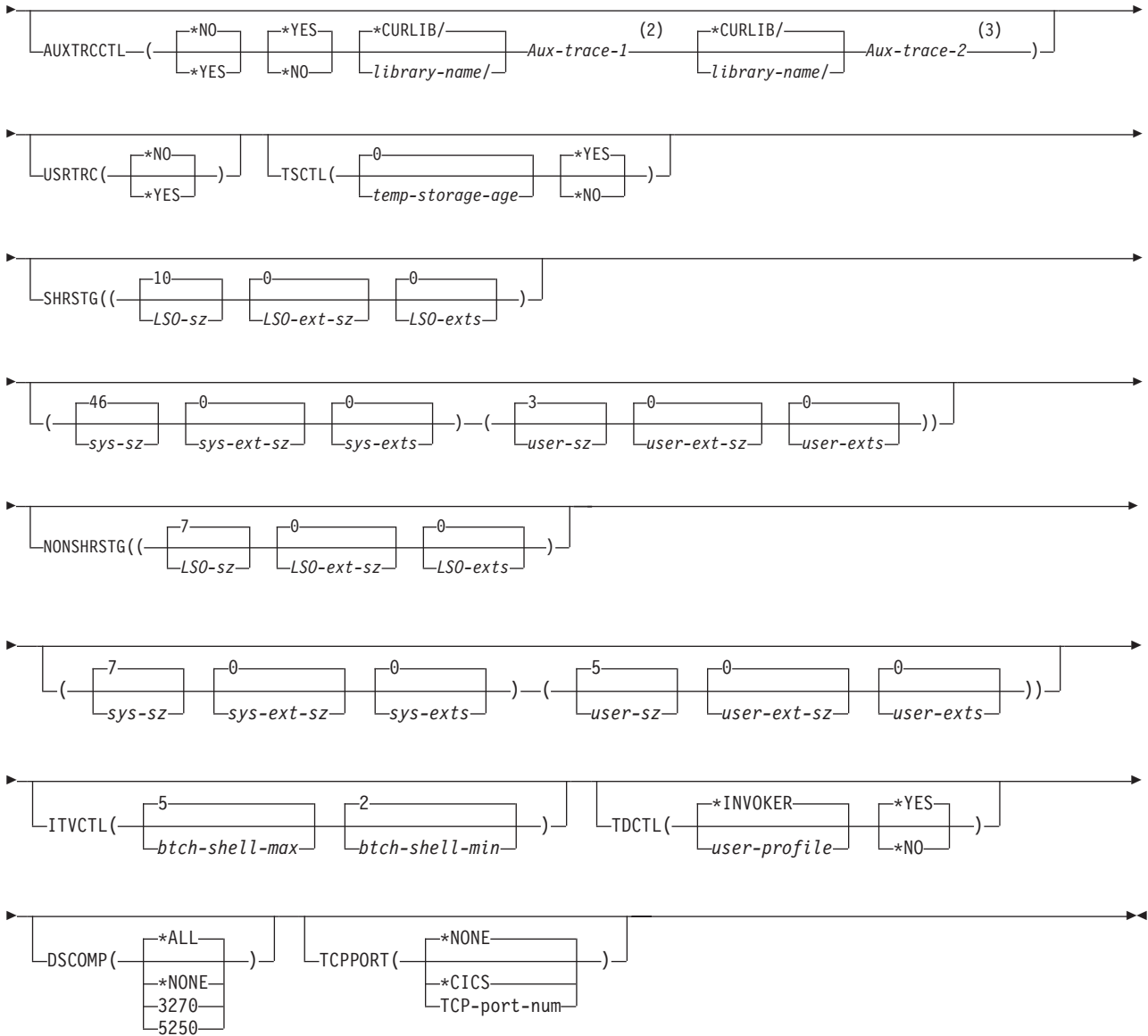
See “Defining the system initialization table (SIT)” on page 61 for information about the SIT supplied with CICS/400.

Using the ADDCICSSIT command

Job: B,I Pgm: B,I REXX: B,I Exec



ADDCICSSIT



Notes:

- 1 All parameters preceding this point can be specified positionally.
- 2 Element 3 (Auxiliary trace file 1) is required when Element 1 (Active) is *YES.
- 3 Element 4 (Auxiliary trace file 2) cannot be the same as Element 3 (Auxiliary trace file 1).

Function

Use the Add CICS/400 System Initialization Table (ADDCICSSIT) command to add the entry to the SIT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this SIT entry is to be added. An ADDCICSSIT command is not valid if a SIT entry already exists for this group.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

GLT library (GLTLIB)

Enter the name of the OS/400 library that contains the group holding the group list table. This name is also used to identify this SIT entry.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

GLT group (GLTGRP)

Enter the name of the group that contains the group list table. All the groups specified in the group list table are used when the CICS/400 control region is started.

Possible values are:

***NONE:** No group list table will be used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Common work area size (WRKARASIZE)

Enter the size of the Common Work Area (CWA) portion of the Common System Area (CSA)

Possible values are:

512: The common work area is 512 bytes.

CWA-size: A number in the range 0 though 3 584.

Application (APPLID)

Enter the VTAM* application identifier for the CICS/400 control region.

Possible values are:

DBDCCICS: This is the default VTAM application identifier to be used with the CICS/400 control region.

VTAM-application-id: The VTAM application identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Date format (DATFORM)

Indicates how the CICS/400 control region system date is formatted when the EXEC CICS FORMATTIME command is issued.

ADDCICSSIT

Possible values are:

***MMDDYY:** Month, day, year.

***DDMMYY:** Day, month, year.

***YYMMDD:** Year, month, day.

CICS system dumps allowed (DUMP)

Indicates whether or not a CICS/400 control region dump can be taken.

Possible values are:

***YES:** CICS/400 control region dumps can be taken.

***NO:** CICS/400 control region dumps cannot be taken.

CICS device processing (DEVCTL)

Enter the terminal control parameters.

Possible values are:

Element 1: CICS device masking

Identifies which characters of the OS/400 device name, as specified in the **CICS device (CICSDEV)** parameter of the TCT, are used to generate the terminal identifier when the user signs on to the CICS/400 control region. The characters are identified by using an offset relative to zero, that is, 1 is the second character of the OS/400 device name.

Note: When one of the characters is not a number, then the character is used in that position of the CICS/400 terminal identifier. For example, terminal mask = 0A5B, OS/400 device name = XYZ1234567, CICS/400 terminal identifier = XA3B.

***DFT:** The CICS/400 terminal mask will consist of the last four nonblank characters of the OS/400 device description.

terminal-mask: The terminal mask may have a maximum length of 4 characters. The characters can be alphanumeric or one of the special characters, \$, @, or #.

Element 2: Autoinstall inactivity limit

The amount of idle time after a CICS/400 shell has ended, that the autoinstall terminal is deleted. The format is *hhmmss*.

0: Delete the autoinstall terminal as soon as the session is ended.

hhmmss: A number in the range 0 through 18 0000, that is up to 18 hours idle time.

Element 3: Device dependent suffixing.

Whether or not device dependent suffixing is enabled.

***DDS:** Allow device dependent suffixing. BMS always tries to load a suffixed version of a map set. The suffix that is used is determined in the following manner. If the transaction uses the alternate screen size, BMS will try to load a map version that has the alternate suffix. If the load fails (or the transaction does not use the alternate screen size), BMS will try to load a version that has a default map suffix. If this fails too, BMS will try to load the unsuffixed version.

***NODDS:** BMS is not to load suffixed versions of map sets. Specifying this option avoids the search for suffixed versions.

File processing (FILECTL)

Enter the parameters used by the file control facility.

Possible values are:

Element 1: Maximum files left open

The maximum number of files that will be left open to a CICS/400 shell at task end.

0: An unlimited number of files can be left open.

max-open: A number in the range 0 through 32 767.

Element 2: Open file inactivity limit

The amount of time in minutes that a file can be left open with no activity.

0: There is no limit to the amount of time that a file can remain open without activity.

minutes: A number in the range 0 through 1440 (24 hours).

Internal trace processing (INTTRCCTL)

Enter the control parameters used by the internal tracing facility.

Possible values are:

Element 1: Maximum trace buffer entries

Maximum number of entries allowed in the internal trace buffer.

125: A maximum of 125 entries are allowed in the internal trace buffer.

trace-buffer-entries: A number in the range 125 through 10 000.

Element 2: Active at startup

Indicates whether or not the internal trace is active, when the CICS/400 control region is started.

*NO: Trace is not active.

*YES: Trace is active.

Element 3: Wrapped when full

Indicates whether or not the internal trace should be wrapped when the internal trace buffer is full.

*YES: The internal trace entries will be wrapped.

*NO: The internal trace is to be stopped.

Auxiliary trace processing (AUXTRCCTL)

Specified the control parameters used by the auxiliary tracing facility.

Possible values are:

Element 1: Active at startup

Indicates whether or not the auxiliary trace is active when the CICS/400 control region is started.

*NO: The trace is not active.

*YES: Trace is active.

Element 2: Automatic switching

Indicates what to do when the auxiliary trace user space is full.

*YES: The trace user space is to be switched.

*NO: The auxiliary trace is to be stopped.

Element 3: Auxiliary trace file 1

The name of the first OS/400 user space object that will be utilized by the auxiliary trace facility.

Possible library values are:

***CURLIB:** The current library for the job that is associated with the CICS/400 control region is used to locate the OS/400 user space object. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the OS/400 user space object is located.

Aux-trace-1: Specify the name of the OS/400 user space object.

Element 4: Auxiliary trace file 2

The name of the second OS/400 user space that will be utilized by the auxiliary trace facility.

Note: This cannot be the same as the OS/400 user space object specified in Element 3 (Auxiliary trace file 1).

Possible library values are:

***CURLIB:** The current library for the job that is associated to the CICS/400 control region is used to locate the OS/400 user space object. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the OS/400 user space object is located.

Aux-trace-2: Specify the name of the OS/400 user space object.

User trace active at startup (USRTRC)

Indicates whether or not user trace entries can be created when the CICS/400 control region is started.

Possible values are:

***NO:** User trace entries will not be created.

***YES:** User trace entries will be created.

Temporary storage processing (TSCTL)

Enter the parameters to be used by the temporary storage facility during either a warm or an emergency restart. See "Starting a control region (STRCICS)" on page 112 for further details.

Possible values are:

Element 1: Age limit

The age limit of temporary storage data used by the temporary storage recovery program during emergency restart of the control region. Data that is older than the specified limit will not be recovered. The value is specified in days.

0: All data is to be recovered.

temp-storage-age: A number in the range 0 through 512.

Element 2: Data recovery

Indicates whether or not the temporary storage queues are recovered during warm or emergency restarts of the CICS/400 control region. See "Setting the STRTYPE parameter" on page 114 for full details of the effects of this element. If no in-doubt units of work are affecting queues, the values have these results:

***YES:** Recover temporary storage queues.

***NO:** Do not recover temporary storage queues.

Shared storage requirements (SHRSTG)

Enter shared storage requirements.

Possible LSO storage requirement values are:

Element 1: Size

Size of the CICS/400 local space object (LSO) stated in KB.

10: CICS/400 LSO will be defined with 10KB of storage.

LSO-sz: A number in the range 10 through 16 000.

Element 2: Extent size

Size of the CICS/400 LSO extents stated in KB.

0: CICS/400 LSO will not be extended when it is filled.

LSO-ext-sz: A number in the range 0 through 16 000.

Element 3: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 LSO.

0: CICS/400 LSO will not be extended when it is filled.

LSO-exts: A number in the range 0 through 32 767.

Possible system storage requirement values are:

Element 4: Size

Size of the CICS/400 system space object stated in KB.

46: CICS/400 system space object will be defined with 46KB of storage.

Sys-sz: A number in the range 46 through 16 000.

Element 5: Extent size

Size of the CICS/400 system space object extents stated in KB.

0: CICS/400 system space object will not be extended with it is filled.

Sys-ext-sz: A number in the range 0 through 16 000.

Element 6: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 system space object.

0: CICS/400 system space object will not be extended when it is filled.

Sys-exts: A number in the range 0 through 32 767.

Possible user storage requirement values are:

Element 7: Size

Size of the CICS/400 user space object stated in KB.

3: CICS/400 user space object will be defined with 3KB of storage.

user-sz: A number in the range 3 through 16 000.

Element 8: Extent size

Size of the CICS/400 user space object extents stated in KB.

0: CICS/400 user space object will not be extended when it is filled.

user-ext-sz: A number in the range 0 through 16 000.

Element 9: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 user space object.

0: CICS/400 user space object will not be extended when it is filled.

ADDCICSSIT

user-exts: A number in the range 0 through 32 767.

Nonshared storage requirements (NONSHRSTG)

Enter the non-shared storage requirements.

Possible LSO storage requirement values are:

Element 1: Size

Size of the CICS/400 LSO stated in KB.

7: CICS/400 LSO will be defined with 7KB of storage.

LSO-sz: A number in the range 7 through 16 000.

Element 2: Extent size

Size of the CICS/400 LSO extents stated in KB.

0: CICS/400 LSO will not be extended when it is filled.

LSO-ext-sz: A number in the range 0 through 16 000.

Element 3: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 LSO.

0: CICS/400 LSO will not be extended when it is filled.

LSO-exts: A number in the range 0 through 32 767.

Possible system storage requirement values are:

Element 4: Size

Size of the CICS/400 system space object stated in KB.

7: CICS/400 system space object will be defined with 7KB of storage.

Sys-sz: A number in the range 7 through 16 000.

Element 5: Extent size

Size of the CICS/400 system space object extents stated in KB.

0: CICS/400 system space object will not be extended when it is filled.

Sys-ext-sz: A number in the range 0 through 16 000.

Element 6: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 system space object.

0: CICS/400 system space object will not be extended when it is filled.

Sys-exts: A number in the range 0 through 32 767.

Possible user storage requirement values are:

Element 7: Size

Size of the CICS/400 user space object stated in KB.

5: CICS/400 user space object will be defined with 5KB of storage.

user-sz: A number in the range 5 through 16 000.

Element 8: Extent size

Size of the CICS/400 user space object extents stated in KB.

0: CICS/400 user space object will not be extended when it is filled.

user-ext-sz: A number in the range 0 through 16 000. **Element 9: Maximum extents**

Maximum number of attempts allowed to extend the CICS/400 user space object.

0: CICS/400 user space object will not be extended when it is filled.

user-exts: A number in the range 0 through 32 767.

Interval control processing (ITVCTL)

Enter the parameters used by interval control facility.

Possible values are:

Element 1: Maximum active CICS shells

The maximum number of batch user shells to be used by interval control, that can be active at the same time.

5: Five batch user shells can be active at the same time.

btch-shell-max: A number in the range 1 through 32 767.

Element 2: Minimum active batch shells

The number of batch user shells to be used by interval control, that will remain active.

2: Two batch user shells will always be active.

btch-shell-min: A number in the range 1 through the value of Element 1 (Maximum active CICS shells).

Transient data information (TDCTL)

Enter the control parameters used by the transient data facility during either a warm or an emergency restart. See “Starting a control region (STRCICS)” on page 112 for further details.

Possible values are:

Element 1: ATI user profile

The OS/400 user profile to be associated with the transaction that is automatically initiated when the transient data trigger level is reached.

***INVOKER**: Use the OS/400 user profile that is associated with the shell that caused the transient data trigger level to be reached.

user-profile: A user profile name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Element 2: Data recovery

Indicates whether or not the transient data queues are recovered during a warm or an emergency restart of the CICS/400 control region. See for full details of the effects of this element. If no in-doubt units of work are affecting queues, the values have these results:

***YES**: Recover transient data queues.

***NO**: Do not recover transient data queues.

Data stream compression (DSCOMP)

Specifies, subject to individual terminal definitions in the TCT, whether terminal output data is to be compressed. See “Datastream compression” on page 274 for further details.

Possible values are:

***ALL**: Output data is compressed for every terminal that has a specification of **DSCOMP(*SITVAL)** in its TCT definition.

***NONE**: No terminal output data is compressed.

3270: Output data is compressed for every 3270 terminal that has a specification of **DSCOMP(*SITVAL)** in its TCT definition.

Output data is not compressed for 5250 terminals.

5250: Output data is compressed for every 5250 terminal that has a specification of **DSCOMP(*SITVAL)** in its TCT definition.

ADDCICSSIT

Output data is not compressed for 3270 terminals.

TCP/IP port number (TCPPORT)

Specifies whether the CICS TCP/IP Listener should be started at Control Region startup, and if so, the TCP port that CICS Clients will use to connect to the listener.

Possible values are:

***NONE:** No CICS TCP/IP Listener is to be used.

***CICS:** The CICS TCP/IP Listener is started at Control Region startup to allow CICS Clients to connect using TCP/IP with the port number specified in the IBM-CICS entry of the Service Table.

TCP port number: A number in the range of 1 through 65535. When a value is specified for this parameter, the CICS TCP/IP Listener is started at Control Region startup to allow CICS Clients to connect using TCP/IP with the specified port number.

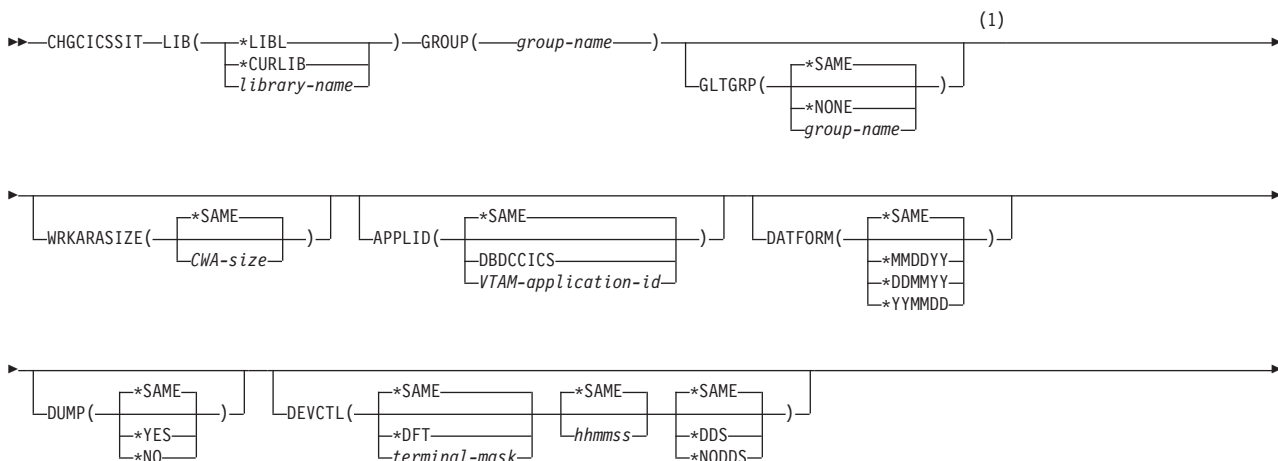
Examples

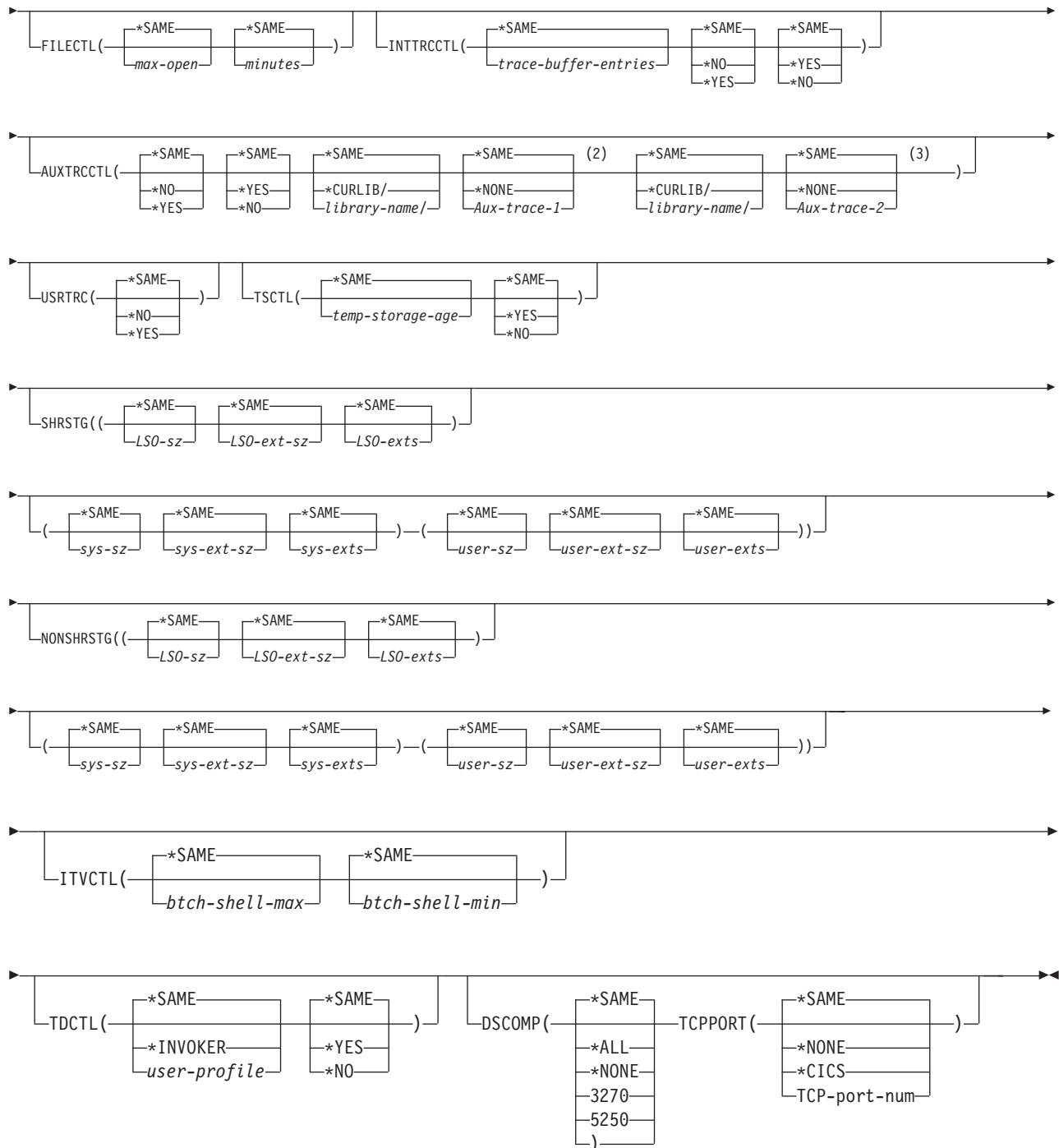
```
ADDCICSSIT LIB(CICSWORK) GROUP(ACCT)
  GLTLIB(CICSWORK) GLTGRP(ACCT)
  WRKARASIZE(100)
  DUMP(*NO)
  DEVCTL(3457)
  INTRCCTL(7000 *YES *NO)
  AUXTRCCTL(*YES *NO CICSWORK/SAMPTRACE1 CICSWORK/SAMPTRACE2)
  USRTRC(*YES)
  TSCTL(0 *NO)
  SHRSTG((400 20 10) (400 20 10) (400 20 10))
  NONSHRSTG((200 20 10) (200 20 10) (200 20 10))
  TDCTL(*INVOKER *NO)
```

This command adds the SIT entry called CICSWORK to group ACCT in OS/400 library CICSWORK. Explanations of the parameter values are given in Table 7 on page 62. All groups that are specified in the GLT located in group ACCT in the OS/400 library ACCT are installed at runtime.

Using the CHGCICSSIT command

Job: B,I Pgm: B,I REXX: B,I Exec



**Notes:**

- 1 All parameters preceding this point can be specified positionally.
- 2 Element 3 (Auxiliary trace file 1) is required when Element 1 (Active) is *YES.
- 3 Element 4 (Auxiliary trace file 2) cannot be the same as Element 3 (Auxiliary trace file 1).

Function

Use the Change CICS/400 System Initialization Table (CHGCICSSIT) command to change the entry in the SIT.

Note: It should be understood that changes made to the SIT can radically affect the performance and runtime characteristics of your CICS system. Changes that are made to these entries should be well understood before they are put into a runtime environment.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the SIT entry to be changed.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

GLT group (GLTGRP)

Enter the name of the group that contains the group list table. All the groups specified in the group list table will be used when starting the CICS/400 control region.

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

***NONE:** No group list table will be used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Common work area size (WRKARASIZE)

Enter the size of the Common Work Area (CWA) portion of the Common System Area (CSA)

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

512: The common work area is 512 bytes.

CWA-size: A number in the range 0 through 3584.

Application (APPLID)

Enter the VTAM application identifier for the CICS/400 control region.

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

DBDCCICS: This is the default VTAM application identifier to be used with the CICS/400 control region.

VTAM-application: The VTAM application identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Date format (DATFORM)

Indicates how the CICS/400 control region system date is formatted when the EXEC CICS FORMATTIME command is issued.

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

***MMDDYY:** Month, day, year.

***DDMMYY:** Day, month, year.

***YYMMDD:** Year, month, day.

CICS system dumps allowed (DUMP)

Indicates whether or not a CICS/400 control region dump can be taken.

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

***YES:** CICS/400 control region dumps can be taken.

***NO:** CICS/400 control region dumps cannot be taken.

CICS device processing (DEVCTL)

Enter the parameters used by the terminal control facility.

Possible values are:

Element 1: CICS device masking

Identifies which characters of the OS/400 device name are used to generate the CICS/400 terminal when the user signs on to the CICS/400 control region. The characters are identified by using an offset relative to zero, that is 1 is the second character of the OS/400 device name.

Note: When one of the characters is not a number, then the character will be used in that position of the terminal identifier. For example, terminal mask = 0A5B, OS/400 device name = XYZ1234567, CICS/400 terminal identifier = XA3B.

***SAME:** Keep the value currently specified in the SIT entry.

***DFT:** The CICS/400 terminal mask will consist of the last four nonblank characters of the OS/400 device description.

terminal-mask: The maximum length is four characters. The characters can be alphanumeric or one of the special characters, \$, @, or #.

Element 2: Autoinstall inactivity limit

The amount of idle time after a CICS/400 shell has ended, that the CICS/400 autoinstall terminal is deleted. The format is *hhmmss*.

***SAME:** Keep the value currently specified in the SIT entry.

hhmmss: A number in the range 0 through 18 0000. When the value specified is 0, then delete the autoinstall terminal as soon as the session is ended.

Element 3: Device dependent suffixing.

Whether or not device dependent suffixing is enabled.

***SAME:** Keep the value currently specified in the SIT entry.

***DDS:** Allow device dependent suffixing. BMS always tries to load a suffixed version of a map set. The suffix that is used is determined in the following manner. If the transaction uses the alternate screen size, BMS will try to load a map version that has the alternate suffix. If the load fails (or

the transaction does not use the alternate screen size), BMS will try to load a version that has a default map suffix. If this fails too, BMS will try to load the unsuffixed version.

***NODDS:** BMS is not to load suffixed versions of map sets. Specifying this option avoids the search for suffixed versions.

File processing (FILECTL)

Enter the parameters used by the file control facility.

Possible values are:

Element 1: Maximum files left open

The maximum number of files that are left open to a CICS/400 shell when a task ends.

***SAME:** Keep the value currently specified in the SIT entry.

max-open: A number in the range 0 through 32 767. When the value specified is 0, then an unlimited number of CICS/400 files can be left open.

Element 2: Open file inactivity limit

The amount of time in minutes that a file can be left open with no activity.

***SAME:** Keep the value currently specified in the SIT entry.

minutes: A number in the range 0 through 1440 (24 hours). When the value specified is 0, then there is no limit to the amount of time that a file can remain open without activity.

Internal trace processing (INTTRCCTL)

Enter the control parameters used by the internal tracing facility.

Possible values are:

Element 1: Maximum trace buffer entries

The maximum number of entries allowed in the internal trace buffer.

***SAME:** Keep the value currently specified in the SIT entry.

trace-buffer-entries: A number in the range 125 through 10 000.

Element 2: Active at startup

Indicates whether or not the internal trace is active, when the CICS/400 control region is started.

***SAME:** Keep the value currently specified in the SIT entry.

***NO:** Trace is not active.

***YES:** Trace is active.

Element 3: Wrapped when full

Indicates whether or not the internal trace should be wrapped when the internal trace buffer is full.

***SAME:** Keep the value currently specified in the SIT entry.

***YES:** The internal trace entries will be wrapped.

***NO:** The internal trace is to be stopped.

Auxiliary trace processing (AUXTRCCTL)

Enter the control parameters used by the auxiliary tracing facility in the CICS/400 control region.

Possible values are:

Element 1: Active at startup

Indicates whether or not the auxiliary trace is active, when the CICS/400 control region is started.

***SAME:** Keep the value currently specified in the SIT entry.

***NO:** The trace is not active.

***YES:** The trace is active.

Element 2: Automatic switching

Indicates what to do when the auxiliary trace user space is full.

***SAME:** Keep the value currently specified in the SIT entry.

***YES:** The trace user space is to be switched.

***NO:** The auxiliary trace is to be stopped.

Element 3: Auxiliary trace file 1

The name of the first OS/400 user space object that will be used by the auxiliary trace facility.

Possible library values are:

***SAME:** Keep the value currently specified in the SIT entry.

***CURLIB:** The current library for the job that is associated to the CICS/400 control region is used to locate the OS/400 user space object. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the OS/400 user space object is located.

Possible file name values are:

***SAME:** Keep the value currently specified in the SIT entry.

***NONE:** There is no OS/400 user space utilized by the auxiliary trace facility.

Aux-trace-1: Specify the name of the OS/400 user space object.

Element 4: Auxiliary trace file 2

The name of the second OS/400 user space that will be used by the auxiliary trace facility.

Note: This cannot be the same as the OS/400 user space object specified in Element 3 (Auxiliary trace file 1).

Possible library values are:

***SAME:** Keep the value currently specified in the SIT entry.

***CURLIB:** The current library for the job that is associated to the CICS/400 control region is used to locate the OS/400 user space object. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the OS/400 user space object is located.

Possible file name values are:

***SAME:** Keep the value currently specified in the SIT entry.

***NONE:** There is no OS/400 user space utilized by the auxiliary trace facility.

Aux-trace-2: Specify the name of the OS/400 user space object.

User trace active at startup (USRTRC)

Indicates whether or not user trace entries can be created when the CICS/400 control region is started.

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

***NO:** User trace entries will not be created.

***YES:** User trace entries will be created.

Temporary storage processing (TSCTL)

Enter the control parameters used by the temporary storage facility.

Possible values are:

Element 1: Age limit

The age limit of temporary storage data used by the temporary storage recovery program during emergency restart of control region. Data that is older than the specified limit will not be recovered. The value is specified in days.

***SAME:** Keep the value currently specified in the SIT entry.

temp-storage-age: A number in range 0 through 512. When the value specified is 0, then all data is to be recovered.

Element 2: Data recovery

Indicates whether or not the temporary storage queues are recovered during warm or emergency restarts of the CICS/400 control region. See "Setting the STRTYPE parameter" on page 114 for full details of the effects of this element. If no in-doubt units of work are affecting queues, the values have these results:

***SAME:** Keep the value currently specified in the SIT entry.

***YES:** Recover temporary storage queues.

***NO:** Do not recover temporary storage queues.

Shared storage requirements (SHRSTG)

Enter the shared storage requirements.

Possible LSO storage requirement values are:

Element 1: Size

Size of the CICS/400 local space object (LSO) stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

LSO-sz: A number in the range 10 through 16 000.

Element 2: Extent size

Size of the CICS/400 LSO extents stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

LSO-ext-sz: A number in the range 0 through 16 000.

Element 3: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 LSO.

***SAME:** Keep the value currently specified in the SIT entry.

LSO-exts: A number in the range 0 through 32 767.

Possible system storage requirement values are:

Element 4: Size

Size of the CICS/400 system space object stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

Sys-sz: A number in the range 46 through 16 000.

Element 5: Extent size

Size of the CICS/400 system space object extents stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

Sys-ext-sz: A number in the range 0 through 16 000.

Element 6: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 system space object.

***SAME:** Keep the value currently specified in the SIT entry.

Sys-exts: A number in the range 0 through 32 767.

Possible user storage requirement values are:

Element 7: Size

Size of the CICS/400 user space object stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

user-sz: A number in the range 3 through 16 000.

Element 8: Extent size

Size of the CICS/400 user space object extents stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

user-ext-sz: A number in the range 0 through 16 000.

Element 9: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 user space object.

***SAME:** Keep the value currently specified in the SIT entry.

user-exts: A number in the range 0 through 32 767.

Nonshared storage requirements (NONSHRSTG)

Enter the non-shared storage requirements.

Possible LSO storage requirement values are:

Element 1: Size

Size of the CICS/400 local space object (LSO) stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

LSO-sz: A number in the range 7 through 16 000.

Element 2: Extent size

Size of the CICS/400 LSO extents stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

LSO-ext-sz: A number in the range 0 through 16 000.

Element 3: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 LSO.

***SAME:** Keep the value currently specified in the SIT entry.

LSO-exts: A number in the range 0 through 32 767.

Possible system storage requirement values are:

Element 4: Size

Size of the CICS/400 system space object stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

Sys-sz: A number in the range 7 through 16 000.

Element 5: Extent size

Size of the CICS/400 system space object extents stated in KB.

***SAME:** Keep the value currently specified in the SIT entry.

Sys-ext-sz: A number in the range 0 through 16 000.

Element 6: Maximum extents

Maximum number of attempts allowed to extend the CICS/400 system space object.

***SAME:** Keep the value currently specified in the SIT entry.

Sys-exts: A number in the range 0 through 32 767.

Possible user storage requirement values are:

Interval control processing (ITVCTL)

Enter the parameters used by the interval control facility.

Possible values are:

Element 1: Maximum active CICS shells

The maximum number of batch user shells to be used by interval control, that can be active at the same time.

***SAME:** Keep the value currently specified in the SIT entry.

btch-shell-max: A number in the range 1 through 32 767.

Element 2: Minimum active CICS shells

The number of batch user shells to be used by interval control, that will remain active.

***SAME:** Keep the value currently specified in the SIT entry.

btch-shell-min: A number in the range 1 through the value of Element 1 (Maximum active CICS shells).

Transient data information (TDCTL)

Enter the control parameters used by transient data facility.

Possible values are:

Element 1: ATI user profile

The OS/400 user profile to be associated with the transaction that is automatically initiated when the transient data trigger level is reached.

***SAME:** Keep the value currently specified in the SIT entry.

***INVOKER:** Use the OS/400 user profile that is associated with the CICS/400 shell that caused the transient data trigger level to be reached.

user-profile: The user profile name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Element 2: Data recovery

Indicates whether or not transient data queues are recovered during either a warm or an emergency restart of the CICS/400 control region. See "Setting the STRTYPE parameter" on page 114 for full details of the effects of this element. If no in-doubt units of work are affecting queues, the values have these results:

***SAME:** Keep the value currently specified in the SIT entry.

***YES:** Recover transient data queues.

***NO:** Do not recover transient data queue.

Data stream compression (DSCOMP)

Specifies, subject to individual terminal definitions in the TCT, whether terminal input data is to be compressed. See “Datastream compression” on page 274 for further details.

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

***ALL:** Output data is compressed for every terminal that has a specification of **DSCOMP(*SITVAL)** in its TCT definition.

***NONE:** No terminal output data is compressed.

3270: Output data is compressed for every 3270 terminal that has a specification of **DSCOMP(*SITVAL)** in its TCT definition.

Output data is not compressed for 5250 terminals.

5250: Output data is compressed for every 5250 terminal that has a specification of **DSCOMP(*SITVAL)** in its TCT definition.

Output data is not compressed for 3270 terminals.

TCP/IP port number (TCPPORT)

Specifies whether the CICS TCP/IP Listener should be started at Control Region startup, and if so, the TCP port that CICS Clients will use to connect to the listener.

Possible values are:

***SAME:** Keep the value currently specified in the SIT entry.

***NONE:** No CICS TCP/IP Listener is to be used.

***CICS:** The CICS TCP/IP Listener is started at Control Region startup to allow CICS Clients to connect using TCP/IP with the port number specified in the IBM-CICS entry of the Service Table.

TCP port number: A number in the range of 1 through 65535. When a value is specified for this parameter, the CICS TCP/IP Listener is started at Control Region startup to allow CICS Clients to connect using TCP/IP with the specified port number.

Examples

```
CHGCICSSIT LIB(SAMPLE1) GROUP(ACCT)
  GLTLIB(SAMPLE1) GLTGRP(ACCT)
  WRKARASIZE(512)
  APPLID(ACCTAPPL)
  DATFORM(*DDMMYY)
  DUMP(*YES)
  DEVCTL(0235 000020)
  FILECTL(3 15)
  INTRCCTL(1000 *YES *YES)
  AUXTRCCTL(*NO *YES SAMPLE1/AUXTRACE1
  SAMPLE1/AUXTRACE2)
  USRTRC(*NO)
  TSCTL(2 *YES)
  SHRSTG((400 20 10) (400 20 10) (400 20 10))
  NONSHRSTG((200 20 10) (200 20 10) (200 20 10))
  ITVCTL(5 2)
  TDCTL(*INVOKER *NO)
```

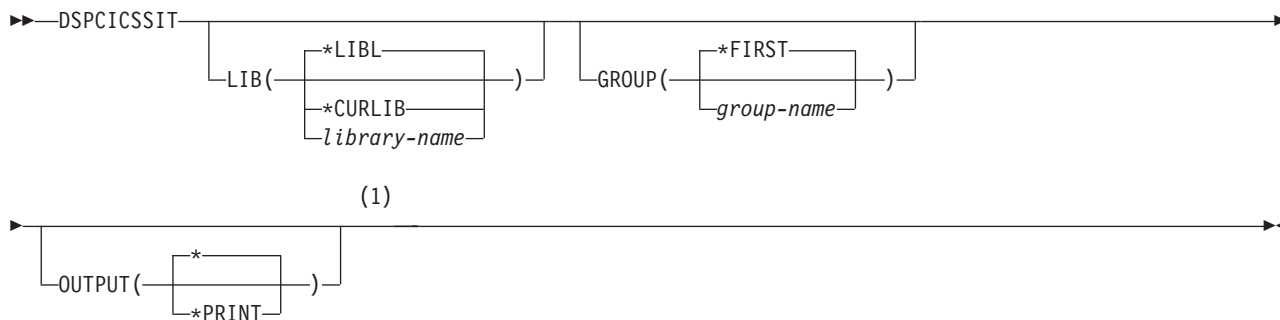
This command changes the shared and non-shared storage requirements for the SIT entry called SAMPLE1 in group ACCT in OS/400 library SAMPLE1.

Explanations of the parameter values are given in Table 7 on page 62. All groups that are specified in the GLT located in group ACCT in the OS/400 library ACCT are installed at runtime.

DSPCICSSIT

Using the DSPCICSSIT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 System Initialization Table (DSPCICSSIT) command to display a SIT entry. You can only view this entry; you can neither make changes to it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the SIT entry to be displayed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*****: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT:** The output is printed with the job spool output.

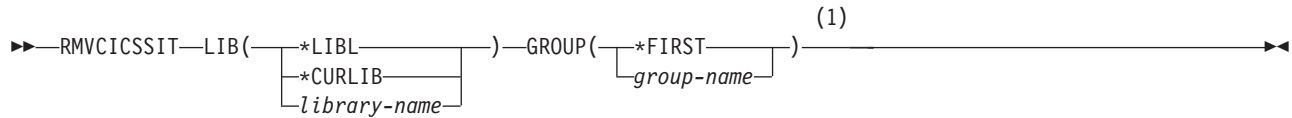
Examples

```
DSPCICSSIT LIB(SAMPLE1) GROUP(ACCT)
```


This command displays the SIT entry located in group ACCT in OS/400 library SAMPLE1.

Using the RMVICSSIT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 System Initialization Table (RMVICSSIT) command to remove a SIT entry from a group.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the SIT entry to be removed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

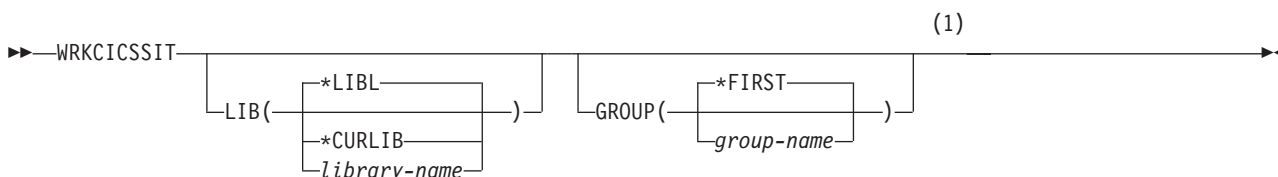
Examples

```
RMVICSSIT LIB(SAMPLE1) GROUP(ACCT)
```

This command removes the SIT entry located in group ACCT in OS/400 library SAMPLE1.

Using the WRKCICSSIT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 System Initialization Table (WRKCICSSIT) command to list the SIT entry. You can change, remove, copy, or display the entry.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the SIT entry to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Examples

```
WRKCICSSIT LIB(SAMPL1) GROUP(ACCT)
```

This command lists the SIT entry located in group ACCT in OS/400 library SAMPLE1.

Managing system resource definitions

The terminal control system table is used to specify the characteristics of remote CICS systems and details of connections between the local and remote systems. An entry in this table:

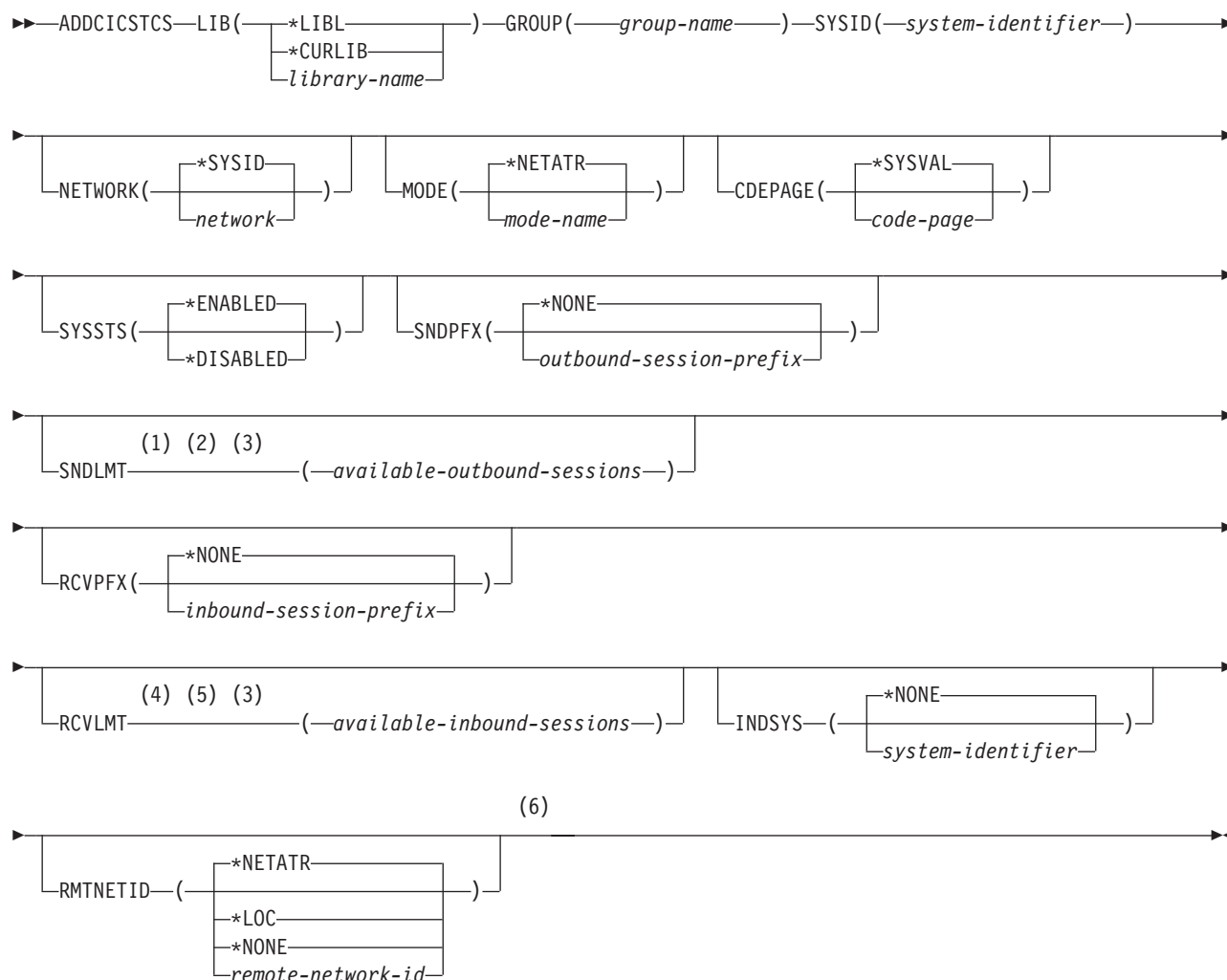
- Defines the link to a remote system. This can be used by local applications for function shipping, distributed program link (DPL), or distributed transaction processing. It can also be used by the system for transaction routing.
- Can be used in the SYSID option of other table entries to identify the system on which a remote resource resides.

- Can be used in the SYSID option of EXEC CICS commands to identify the location of a remote resource.

See “Device considerations” on page 100 for guidance information.

Using the ADDCICSTCS command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The SNDLMT parameter is not valid when SNDPFX(*NONE) is specified.
- 2 The SNDLMT parameter is required when SNDPFX(*NONE) is not specified and the SNDPFX value is not the same as the RCVPFX value.
- 3 SNDLMT must be 1 and RCVLMT must be 0 when the SNDPFX value is the same as the RCVPFX value.
- 4 The RCVLMT parameter is not valid when RCVPFX(*NONE) is specified.
- 5 The RCVLMT parameter is required when RCVPFX(*NONE) is not specified and the SNDPFX value is not the same as the RCVPFX value.
- 6 All parameters preceding this point can be specified positionally.

Function

Use the Add CICS/400 Terminal Control System Table (ADDCICSTCS) command to add an entry to the TCS.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

The name of the group to which the remote TCS entry is to be added.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric \$, @, or #.

CICS system (SYSID)

The system identifier used to identify the remote CICS system that can communicate with the CICS/400 control region. This is the name of the TCS entry.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

Network (NETWORK)

The network name used to identify the other CICS system. This has to be the same as the application identifier parameter (APPLID) specified in the CICS system initialization table used to start up the CICS system.

Possible values are:

***SYSID:** The system identifier suffixed with four blanks will be used as the network name.

network: The network may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Mode (MODE)

The name that is passed as the mode name.

Possible values are:

***NETATR:** The system identifier will use the OS/400 mode name. This is not supported for client entries.

mode-name: The mode name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Code page (CDEPAGE)

The code page to be used for conversion of data.

Possible values are:

***SYSVAL:** Use the OS/400 system code page.

code-page: The code page must be numeric, greater than or equal to 1 and less than or equal to 65 535.

Status (SYSSTS)

Indicates whether the system identifier is available for use.

Possible values are:

***ENABLED:** The system identifier can be used.

***DISABLED:** The system identifier cannot be used.

Outbound session prefix (SNDPFX)

The prefix to be used as the first part of a conversation identifier for outbound sessions (line connections) to establish unique session names.

Note: When the SNDPFX has the same value as the RCVVPFX, the SNDLMT is set to 1 and the RCVLMT is set to 0.

Possible values are:

***NONE:** There are no outbound sessions.

outbound-session-prefix: The outbound session prefix may have a maximum length of 2 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The second character can be alphanumeric or \$, @, or #.

Available outbound sessions (SNDLMT)

Identifies the number of outbound sessions (line connections) that can be active at any one time.

available-outbound-sessions: The available-outbound-sessions must be numeric, greater than or equal to 0 and less than or equal to 999.

Note: When SNDPFX is 1 character, the value should not be greater than 999; otherwise the value should not be greater than 99.

Inbound session prefix (RCVVPFX)

The prefix to be used as the first part of a conversation identifier for inbound sessions (line connections) to establish unique session names.

Note: When the SNDPFX has the same value as the RCVVPFX, then the SNDLMT is set to 1 and the RCVLMT is set to 0.

Possible values are:

***NONE:** There are no inbound sessions.

inbound-session-prefix: The inbound session prefix may have a maximum length of 2 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The second character can be alphanumeric or \$, @, or #.

Available inbound sessions (RCVLMT)

Indicates the number of inbound sessions (line connections) that can be active at any one time.

available-inbound-sessions: The available-inbound-sessions must be numeric, greater than or equal to 0 and less than or equal to 999.

ADDCICSTCS

Note: When RCVPFX is 1 character, the value should be less than or equal to 999; otherwise the value should be less than or equal to 99.

Indirect CICS system (INDSYS)

Indicates whether the remote system communicates with the control region through a second system.

Possible values are:

***NONE:** The remote system can communicate directly with the control region.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote network indicator (RMTNETID)

Enter the remote network identifier to be used with the other CICS system.

Possible values are:

***NETATR:** The remote network identifier specified in the network attributes is used. If this is specified for a client entry, the client must be on the same network as the iSeries to which it connects.

***LOC:** Any remote network identifier for the other CICS system may be used. If several remote network identifiers are associated with the other CICS system, the system automatically selects the remote network identifier.

***NONE:** No remote network identifier is used.

remote-network-id: The remote network identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

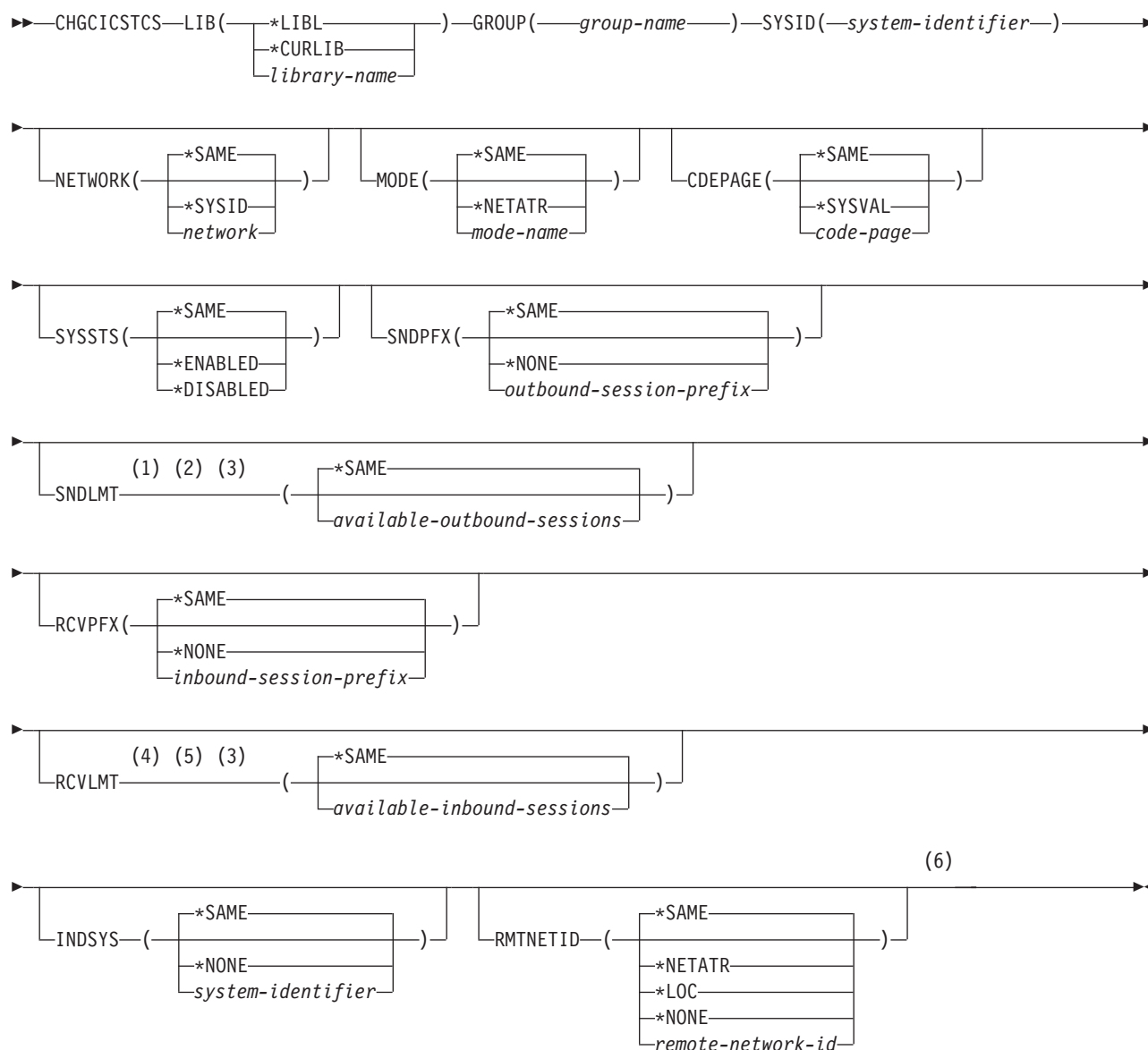
Examples

```
ADDCICSTCS LIB(CICSWORK) GROUP(ACCT)
  SYSID(SYS1) NETWORK(CICSSYS1) SNDFPX(S1)
  SNDLMT(3) RCVPFX(I1) RCVLMT(3)
```

This command adds a TCS entry for a remote system called SYS1 to group ACCT in OS/400 library CICSWORK. The network name associated with the CICS/400 control region is CICSSYS1. By default, the entry is enabled. A maximum of three send and three receive sessions is allowed.

Using the CHGCICSTCS command

Job: B,I Pgm: B,I REXX: B,I Exec

**Notes:**

- 1 The SNLMT parameter is not valid when SNDPFX(*NONE) is specified.
- 2 The SNLMT parameter is required when SNDPFX(*NONE) is not specified and the SNDPFX value is not the same as the RCVPFX value.
- 3 SNLMT must be 1 and RCVLMT must be 0 when the SNDPFX value is the same as the RCVPFX value.
- 4 The RCVLMT parameter is not valid when RCVPFX(*NONE) is specified.
- 5 The RCVLMT parameter is required when RCVPFX(*NONE) is not specified and the SNDPFX value is not the same as the RCVPFX value.
- 6 All parameters preceding this point can be specified positionally.

Function

Use the Change CICS/400 Terminal Control System Table (CHGCICSTCS) command to change an entry in the TCS.

Required parameters**Library (LIB)**

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

The name of the group containing the remote TCS entry to be changed.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS system (SYSID)

The system identifier used to identify the remote CICS system that can communicate with the CICS/400 control region. This is the name of the TCS entry.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters**Network (NETWORK)**

The network name used to identify the other CICS system. This has to be the same as the application identifier parameter (APPLID) specified in the CICS system initialization table used to start up the CICS system.

Possible values are:

***SAME:** The value currently specified in the TCS table entry will remain the same.

***SYSID:** The system identifier suffixed with four blanks will be used as the network name.

network: The network may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Mode (MODE)

The name that is passed as the mode name.

Possible values are:

***SAME:** The value currently specified in the TCS table entry will remain the same.

***NETATR:** The system identifier will use the OS/400 mode name. This is not supported for client entries.

mode-name: The mode name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Code page (CDEPAGE)

The code page to be used for conversion of data.

Possible values are:

***SAME:** The value currently specified in the TCS table entry will remain the same.

***SYSVAL:** Use the OS/400 system code page.

code-page: The code page must be numeric, greater than or equal to 1 and less than or equal to 65535.

Status (SYSSTS)

Indicates whether the system identifier is available for use.

Possible values are:

***SAME:** The value currently specified in the TCS table entry will remain the same.

***ENABLED:** The system identifier can be used.

***DISABLED:** The system identifier cannot be used.

Outbound session prefix (SNDPFX)

The prefix to be used as the first part of a conversation identifier for outbound sessions (line connections) to establish unique session names.

Note: When the SNDPFX has the same value as the value in RCVPFX, then the SNDLMT will be set to 1 and the RCVLMT will be set to 0.

Possible values are:

***SAME:** The value currently specified in the CICS/400 TCS table entry will remain the same.

***NONE:** There are no outbound sessions.

outbound-session-prefix: The outbound session prefix may have a maximum length of 2 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Available outbound sessions (SNDLMT)

Identifies the number of outbound sessions (line connections) that can be active at any one time.

Possible values are:

***SAME:** The value currently specified in the TCS table entry will remain the same.

available-outbound-sessions: The available-outbound-sessions must be numeric, greater than or equal to 0 and less than or equal to 999.

Note: When SNDPFX is 1 character, the value should not be greater than 999; otherwise the value should not be greater than 99.

Inbound session prefix (RCVPFX)

Enter the prefix to be used as the first part of a conversation identifier for inbound sessions (line connections) to establish unique session names.

Note: When the SNDPFX has the same value as the RCVPFX, the SNDLMT is set to 1 and the RCVLMT is set to 0.

Possible values are:

***SAME:** Keep the value currently specified in the TCS table entry.

***NONE:** There are no inbound sessions.

inbound-session-prefix: The inbound session prefix may have a maximum length of 2 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The second character can be alphanumeric or \$, @, or #.

Available inbound sessions (RCVLMT)

Indicates the number of inbound sessions (line connections) that can be active at any one time.

Possible values are:

***SAME:** The value currently specified in the TCS table entry will remain the same.

available-inbound-sessions: The available-inbound-sessions must be numeric, greater than or equal to 1 and less than or equal to 999.

Note: When RCVPFX is 1 character, the value should be less than or equal to 999; otherwise the value should be less than or equal to 99.

Indirect CICS system (INDSYS)

Indicates whether the remote system communicates with the control region through a second system.

Possible values are:

***SAME:** The value currently specified in the TCS table entry will remain the same.

***NONE:** The remote system can communicate directly with the control region.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote network indicator (RMTNETID)

Enter the remote network identifier to be used with the other CICS system.

Possible values are:

***SAME:** Keep the value currently specified for this entry.

***NETATR:** The remote network identifier specified in the network attributes is used. If this is specified for a client entry, the client must be on the same network as the iSeries to which it connects.

***LOC:** Any remote network identifier for the other CICS system may be used. If several remote network identifiers are associated with the other CICS system, the system automatically selects the remote network identifier.

***NONE:** No remote network identifier is used.

remote-network-id: The remote network identifier may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

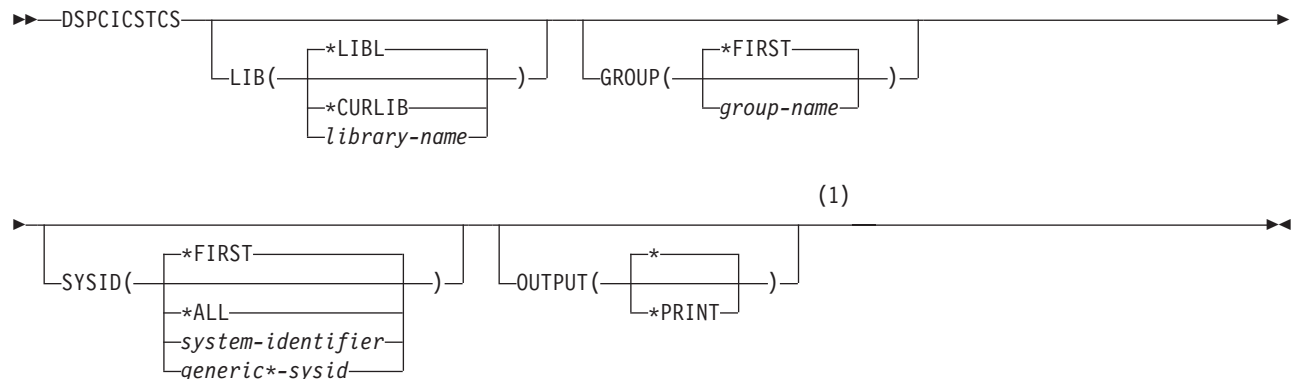
Examples

```
CHGCICSTCS LIB(CICSWORK) GROUP(ACCT)
SYSID(SYS1) SNDLMT(4)
```

This command changes the TCS entry called SYS1 to group ACCT in OS/400 library CICSWORK. The number of receive sessions allowed has been increased to 4.

Using the DSPCICSTCS command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Terminal Control System Table (DSPCICSTCS) command to display a TCS entry. You can only view this entry; you can neither make changes to it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library which contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

The name of the remote group containing the TCS entry to be displayed.

Possible values are:

***FIRST:** No group is specified, the first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS system (SYSID)

The system identifier used to identify the remote CICS system that can communicate with the CICS/400 control region. This is the name of the TCS entry.

Possible values are:

DSPCICSTCS

***FIRST:** Display the first TCS entry.

***ALL:** Display all of the TCS entries.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-sysid:* Specify the generic name of the system identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all entries with a system identifier beginning with the generic name are displayed. If an asterisk is not included with the generic name, the system assumes the value to be the complete system identifier.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*****: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT:** The output is printed with the job spool output.

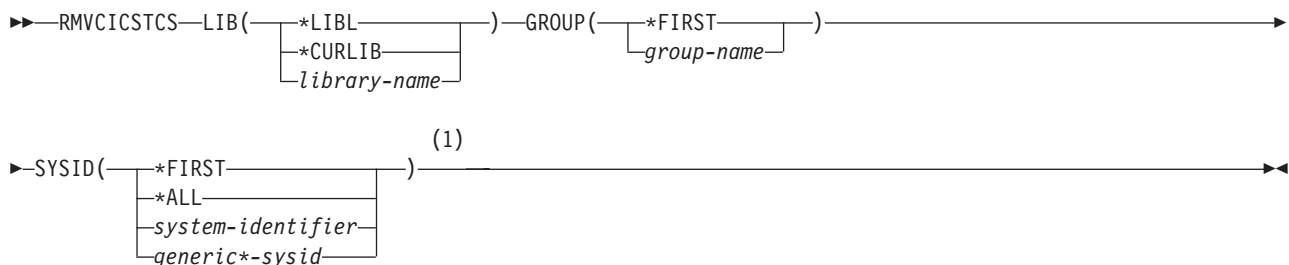
Examples

```
DSPCICSTCS LIB(CICSWORK) GROUP(ACCT) SYSID(*ALL)
```

This command displays all TCS entries located in group ACCT in OS/400 library CICSWORK.

Using the RMVCICSTCS command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Terminal Control System Table (RMVCICSTCS) command to delete an entry from the TCS.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group. Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

The name of the remote group containing the TCS entry to be removed.

Possible values are:

***FIRST:** No group is specified, the first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS system (SYSID)

The system identifier used to identify the remote CICS system that can communicate with the CICS/400 control region. This is the name of the TCS entry.

Possible values are:

***FIRST:** Remove the first TCS table entry.

***ALL:** Remove all of the TCS table entries.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-sysid:* Specify the generic name of the system identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all entries with a system identifier beginning with the generic name are removed. If an asterisk is not included with the generic name, the system assumes the value to be the complete system identifier.

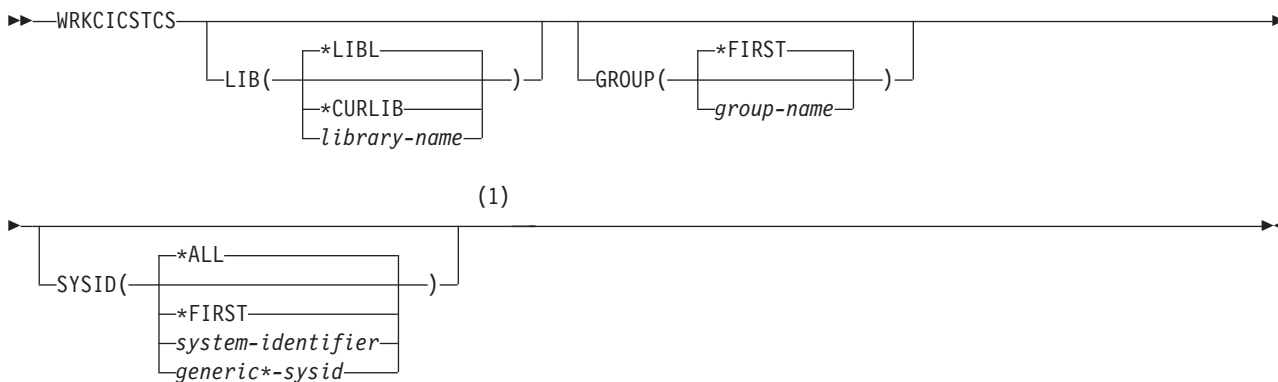
Examples

```
RMVICSTCS LIB(CICSWORK) GROUP(ACCT)
  SYSID(ABC*)
```

This command removes all TCS entries, that start with ABC and end with anything, located in group ACCT in OS/400 library CICSWORK.

Using the WRKCICSTCS command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Terminal Control System Table (WRKCICSTCS) command to list entries in the TCS table. You can change, remove, copy or display entries, or add new ones to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library which contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the TCS entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS system (SYSID)

Enter the name of the TCS table entry to be listed. This is the system identifier used to identify the remote CICS system.

Possible values are:

***ALL:** List all TCS entries.

***FIRST:** List the first TCS entry.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-sysid*: Specify a generic system identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all entries with a system identifier beginning with the generic name are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete system identifier.

Examples

```
WRKICSTCS LIB(CICSWORK) GROUP(ACCT)
```

This command lists all TCS entries located in group ACCT in OS/400 library CICSWORK.

Managing terminal resource definitions

Use the terminal control table (TCT) to define:

- The characteristics of a display terminal or printer, that can be used by transactions running on this system. The terminal may be local or remote.
- Model terminals for automatic installation (autoinstall). Because many terminals have many characteristics in common and their resource definitions would be identical, you can use the TCT to create model definitions. Each model definition may apply to a number of terminals. These models are used by CICS/400 to create the necessary terminal definitions for individual terminals when the system is started.
- A terminal to be used for automatic transaction initiation.

See “Device considerations” on page 100 for guidance information.

Uppercase conversion of terminal input

Transaction IDs and application data can be entered at terminals. In some installations, transaction IDs are case-sensitive; for example, ABCD, abcd, and aBcD could represent three different transactions. Some applications require all input data in uppercase.

Using the interaction of UCTRN options in the TCT and PCT resource definitions, CICS/400 can distinguish between transaction IDs and data, and can apply the correct uppercase conversion to either, neither, or both, as required.

For user data, UCTRN(*YES) in a TCT or PCT entry forces uppercase conversion for all user data associated with the terminal or transaction being defined. UCTRN(*YES) or UCTRN(*TRANID) in the TCT forces uppercase conversion of all transaction IDs entered at the terminal being defined.

To suppress uppercase conversion of data, both the TCT and the PCT must specify UCTRN(*NO).

When both the TCT and PCT specify UCTRN(*YES), a DBCS character is subject to alteration if either its first or second byte has the equivalent code point as that of an SBCS small Latin character in the code page used by CICS. To avoid this, a user

WRKCICSTCS

who would like to allow DBCS data in the application is advised to set UCTRN(*NO) in the PCT and either UCTRN(*NO) or UCTRN(*TRNID) in the TCT.

Table 24 shows the results of the various combinations of UCTRN values in the TCT and PCT.

Table 24. Results of UCTRN settings in TCT and PCT. This table shows when terminal-entered transaction IDs and data are converted to uppercase, as indicated in each box by **Yes** or **No**.

Transaction (PCT)	Terminal (TCT)		
	UCTRN(*YES)	UCTRN(*NO)	UCTRN(*TRANID)
UCTRN(*YES)	Transaction ID: Yes	Transaction ID: No	Transaction ID: Yes
	Data: Yes	Data: Yes	Data: Yes
UCTRN(*NO)	Transaction ID: Yes	Transaction ID: No	Transaction ID: Yes
	Data: Yes	Data: No	Data: No

Notes:

1. In the TCT, UCTRN(*YES) causes all transaction IDs and data entered at the terminal to be converted to uppercase, regardless of the PCT setting for individual transactions.
2. In the PCT, UCTRN(*YES) causes all input data for the transaction to be converted to uppercase, regardless of the TCT setting for individual terminals.
3. In both tables, UCTRN(*NO) indicates that no translation to uppercase is required. In either table, UCTRN(*NO) can be overridden by the UCTRN value in the other table.
4. In the TCT, UCTRN(*TRANID) causes all transaction IDs entered at the terminal to be converted to uppercase. For data, UCTRN(*TRANID) is equivalent to UCTRN(*NO).
5. In the PCT, the UCTRN option relates to data only, and has no effect on conversion of the transaction ID.

Datastream compression

Datastream compression reduces the length of datastreams at a cost in performance¹. In the system initialization table (SIT), the DSCOMP option has four possible values— *ALL, *NONE, 3270, and 5250, which allow compression for all terminals, no terminals, 3270 terminals only, or 5250 terminals only, respectively.

In the terminal control table (TCT), the DSCOMP option gives flexibility at the terminal level. In a 3270 terminal definition, DSCOMP(*SITVAL) allows compression for the terminal if DSCOMP in the SIT specifies *ALL or 3270. In a 5250 terminal definition, DSCOMP(*SITVAL) allows compression for the terminal if DSCOMP in the SIT specifies *ALL or 5250. DSCOMP(*NO) for a terminal prevents compression, regardless of the SIT specification.

Table 25 on page 275 shows the results of all combinations of DSCOMP settings in the SIT and TCT. An X in the **Compress** column, indicates when compression takes place. The table shows that compression occurs only when the TCT entry specifies DSCOMP(*SITVAL) *and* the SIT allows compression for the type of terminal.

1. double-byte character set (DBCS) datastreams are never compressed

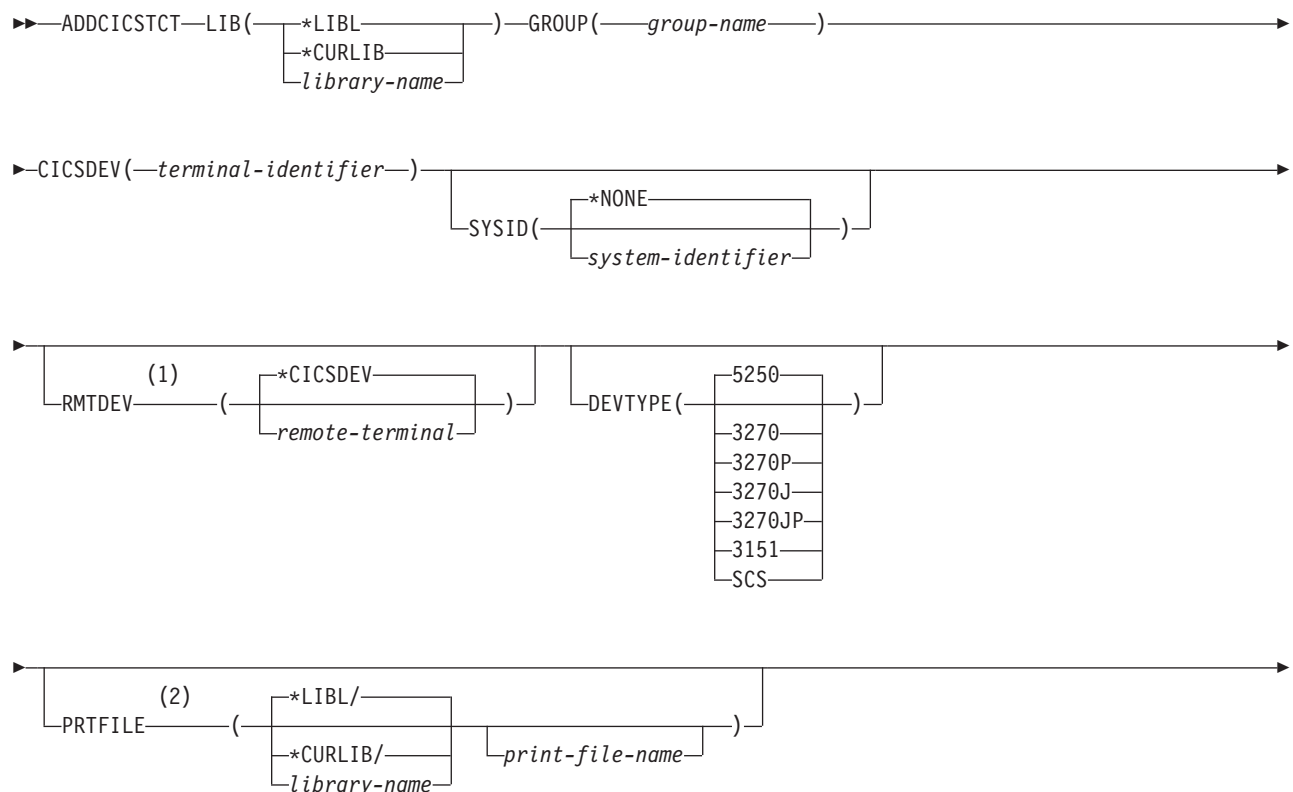
Table 25. Results of DSCOMP settings in TCT and SIT

SIT	3270		5250	
	TCT	Compress	TCT	Compress
ALL	*SITVAL	X	*SITVAL	X
	*NO		*NO	
*NONE	*SITVAL		*SITVAL	
	*NO		*NO	
3270	*SITVAL	X	*SITVAL	
	*NO		*NO	
5250	*SITVAL		*SITVAL	X
	*NO		*NO	

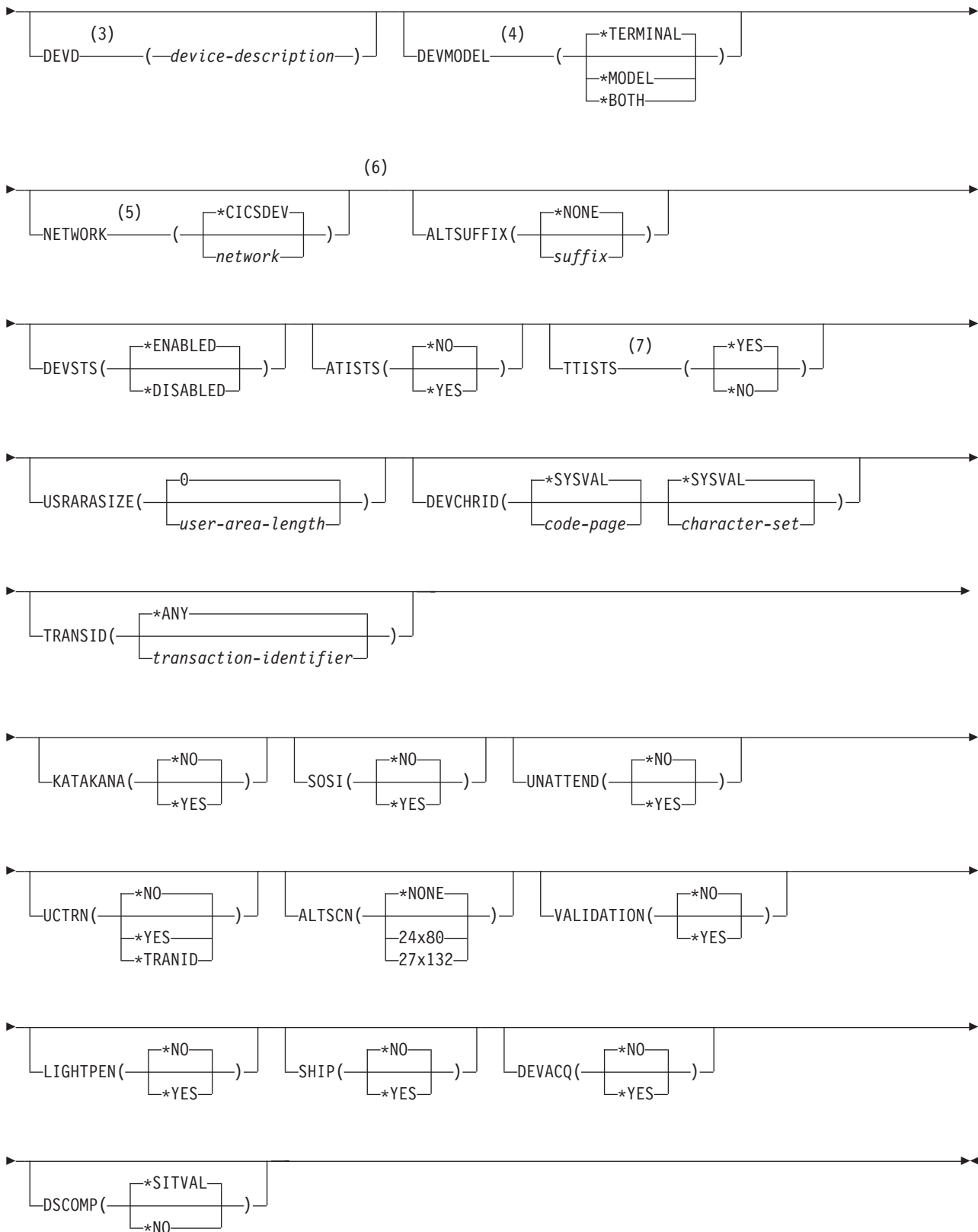
Note: Programming techniques can also be used to reduce the size of the datastream, but may be considered unnecessary if CICS/400 datastream compression is used. If these techniques are employed, they are not affected by the datastream compression applied by the system. For further information, refer to the *CICS for iSeries Application Programming Guide*.

Using the ADDCICSTCT command

Job: B,I Pgm: B,I REXX: B,I Exec



ADDCICSTCT



Notes:

- 1 The RMTDEV parameter is not valid when SYSID(*NONE) is specified.

- 2 The PRTFILE parameter is valid, and required, when DEVTYPE(SCS) is specified.
- 3 The DEVD parameter is valid, and required, when SYSID(*NONE) is specified, DEVTYPE is not SCS, and DEVMODEL is not *MODEL.
- 4 The DEVMODEL parameter is valid only when DEVTYPE(5250), DEVTYPE(3270), DEVTYPE(3270J), or DEVTYPE(3151) is specified.
- 5 The NETWORK parameter is not valid when DEVMODEL(*MODEL) is specified.
- 6 All parameters preceding this point can be specified positionally.
- 7 TTISTS(*NO) is valid only when ATISTS(*YES) is specified.

Function

Use the Add CICS/400 Terminal Control Table (ADDCICSTCT) command to add an entry to the TCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the remote group to which the TCT entry is to be added.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS device (CICSDEV)

The terminal id that will be used to refer to this resource. If this TCT entry is defining a model terminal, this parameter identifies the entry, rather than a specific terminal.

terminal-identifier: The terminal identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters

CICS system (SYSID)

The name of the remote system identifier defined in the TCS Table in which the device is located. This parameter is required only for remote terminals. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***NONE:** The terminal is defined to the same control region in which it is being used.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

ADDCICSTCT

Remote CICS device (RMTDEV)

The name by which the terminal is known in the remote system. This parameter is not valid when SYSID(*NONE) is specified.

Possible values are:

***CICSDEV:** The terminal associated with the TCT is to be used.

remote-terminal: The remote terminal may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Device type (DEVTYPE)

Type of the CICS/400 terminal.

Possible values are:

5250: Terminal supporting 5250 data stream.

3270: Terminal supporting 3270 data stream.

3270P: Printer supporting 3270 data stream.

3270J: Double-byte capable display.

3270JP:
Double-byte capable printer.

3151: ASCII display.

SCS: Printer supporting SCS data stream.

Print file (PRTFILE)

The name of the print spool file that will be used by this terminal. This is only valid when DEVTYPE(SCS) is specified.

Possible values are:

***LIBL:** The library list for the job that is associated to the control region is used to locate the file.

***CURLIB:** The current library for the job that is associated to the control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

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

Device description (DEVDD)

The OS/400 device name that is associated with the terminal. This is only valid when SYSID(*NONE) is specified, DEVTYPE is not SCS, and DEVMODEL is not *MODEL.

device-description: The device description may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Autoinstall model (DEVMODEL)

Indicates whether this terminal can be used as a model to autoinstall terminals. This is only valid when DEVTYPE(5250), DEVTYPE(3270), DEVTYPE(3270J), or DEVTYPE(3151) is specified.

Possible values are:

***TERMINAL:** The terminal cannot be used as a model to autoinstall terminals.

***MODEL:** The terminal can only be used as a model to autoinstall further terminals.

***BOTH:** The terminal can be used as a model to autoinstall further terminals.

Network (NETWORK)

The symbolic network name used to identify the logical unit as it is known throughout the network. The name is supplied to VTAM system definition and is used to build the node initialization block (NIB). This is not valid when DEVMODEL(*MODEL) is specified.

Possible values are:

***CICSDEV:** The terminal suffixed with four blanks will be used as the symbolic network name.

network: The network may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Alternate suffix (ALTSUFFIX)

Enter a number to specify an alternate suffix for this device.

Possible values are:

***NONE:** The device will not have a user defined alternate suffix.

suffix: A number in the range 0 through 9. This numeric suffix will be appended by BMS to map set names, if the transaction uses the alternate screen size (or if the default and alternate screen sizes are the same).

Status (DEVSTS)

Indicates whether the terminal can be used by the control region when the TCT entry is defined to the runtime resource table definition Terminal Control facility.

Possible values are:

***ENABLED:** The terminal can be used by the control region.

***DISABLED:** The terminal cannot be used by the control region.

ATI supported (ATISTS)

Indicates whether the terminal can be used for ATI transactions, or for an ISC session. ISC sessions are for transactions using the terminal as an alternate facility to communicate with another CICS system.

Possible values are:

***NO:** The terminal cannot be used by the control region ATI facility.

***YES:** The terminal can be used by the control region ATI facility.

Transaction entry supported (TTISTS)

Indicates whether transactions may be started at this terminal.

Note: When ATISTS(*NO) is specified, this parameter must be *YES.

Possible values are:

***YES:** Transactions may be started at this terminal.

***NO:** Transactions may not be started at this terminal. Transactions running on this terminal must be started by ATI or by EXEC CICS START commands.

User area size (USRARASIZE)

The length of the user area associated with the terminal.

ADDCICSTCT

Possible values are:

0: No user area will be used.

user-area-length: The user-area-length must be numeric, greater than or equal to 0 and less than or equal to 255.

Character identifier (DEVCHRID)

The code page and character set to be used with the terminal.

Possible values are:

Element 1: Code page

The code page to be used.

***SYSVAL:** Use the OS/400 system code page.

code-page: The code-page must be numeric, greater than or equal to 1 and less than or equal to 65 535.

Element 2: Graphic character set

The graphic character set to be used.

***SYSVAL:** Use the OS/400 system character set.

character-set: The character-set must be numeric, greater than or equal to 1 and less than or equal 65 535.

Transaction (TRANSID)

The transaction identifier defined in the PCT when it is the only transaction that can be run on the terminal.

Possible values are:

***ANY:** Any transaction can be executed on the terminal.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Japanese alphabet supported (KATAKANA)

Indicates whether Katakana support is required. Katakana terminals cannot display mixed case output; uppercase characters appear as uppercase English characters, but lowercase characters appear as Katakana characters. BMS provides automatic uppercase translation of lowercase Latin characters, in BMS fields where the CASE=MIXED option is specified, if Katakana support is required on the terminal where the map is to be displayed.

User written application may query this terminal attribute and provide additional support if required. Possible values are:

***NO:** The terminal does not require Katakana support.

***YES:** The terminal does require Katakana support.

User specified DBCS data (SOSI)

Indicates whether or not the terminal has a mixed EBCDIC/DBCS field capability.

Possible values are:

***NO** The terminal does not have a mixed EBCDIC/DBCS field capability.

***YES:** The terminal has a mixed EBCDIC/DBCS field capability.

Unattended mode (UNATTEND)

Indicates whether or not the mode of operation for the terminal is unattended.

Possible values are:

***NO:** The terminal is to have an attended mode of operation.

***YES:** The terminal is to have an unattended mode of operation.

Auto upper case translation (UCTRN)

Indicates whether or not the terminal is to have uppercase translation done by CICS/400.

Possible values are:

***NO:** CICS/400 translates lowercase characters in data input from this terminal to uppercase only if the transaction requires data in uppercase (UCTRN option in ADDCICSPCT command). CICS/400 never translates lowercase characters in a transaction ID.

***YES:** CICS/400 always translates lowercase characters in input (data or transaction ID) from this terminal to uppercase.

***TRANID:** CICS/400 translates lowercase characters in data input from this terminal to uppercase only if the transaction requires data in uppercase (UCTRN option in ADDCICSPCT command). CICS/400 always translates a transaction ID that contains any lowercase characters to be completely uppercase.

Table 24 on page 274 shows the results of the various combinations of UCTRN values in the TCT and PCT.

Alternate screen size (ALTSCN)

Indicates the alternate screen height and width associated with the terminal.

Possible values are:

***NONE:** No alternate screen size will be used.

24x80: 24 rows and 80 columns will be used as the alternate screen size.

27x132: 27 rows and 132 columns will be used as the alternate screen size.

Validation capability (VALIDATION)

Indicates whether or not the terminal has validation capability. This consists of mandatory fill and mandatory enter.

Possible values are:

***NO:** The terminal does not have validation capability.

***YES:** The terminal has validation capability.

Light pen supported (LIGHTPEN)

Indicates whether or not the terminal has light pen capability.

Possible values are:

***NO:** The terminal does not have light pen capability.

***YES:** The terminal has light pen capability.

Ship to another CICS system (SHIP)

Indicates whether or not the terminal definition is to be shipped to the other CICS system.

Possible values are:

***NO:** The terminal definition will not be shipped to the other CICS system.

***YES:** The terminal definition will be shipped to the other CICS system.

ATI acquire (DEVACQ)

Indicates whether or not the terminal is to be acquired by transactions that are initiated automatically by the control region.

Possible values are:

ADDCICSTCT

***NO:** The control region ATI facility will wait to initiate the transaction, until the terminal is in an acquire state.

***YES:** The control region ATI facility will acquire the terminal if it is not already in an acquire state, before initiating the transaction.

Datastream compression (DSCOMP)

Indicates whether or not a datastream sent to this terminal is to be compressed.

Possible values are:

***SITVAL:** Data output to this terminal is compressed if the DSCOMP option in the SIT (system initialization table) specifies the type of this terminal (3270 or 5250) or *ALL.

***NO:** Data output to this terminal is never compressed.

Table 25 on page 275 shows the results of the various combinations of DSCOMP values in the TCT and SIT.

Examples

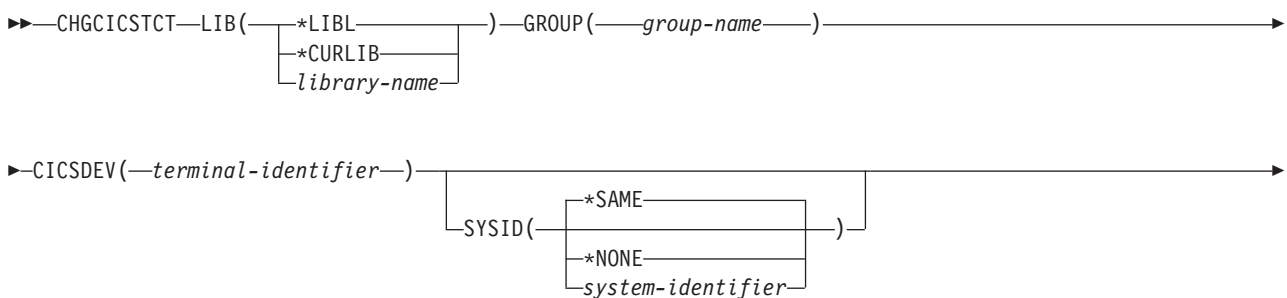
```
ADDCICSTCT LIB(QCICSSAMP) GROUP(ACCT)
CICSDEV(AP01)
DEVD(DEVICE1) DEVMODEL(*BOTH)
```

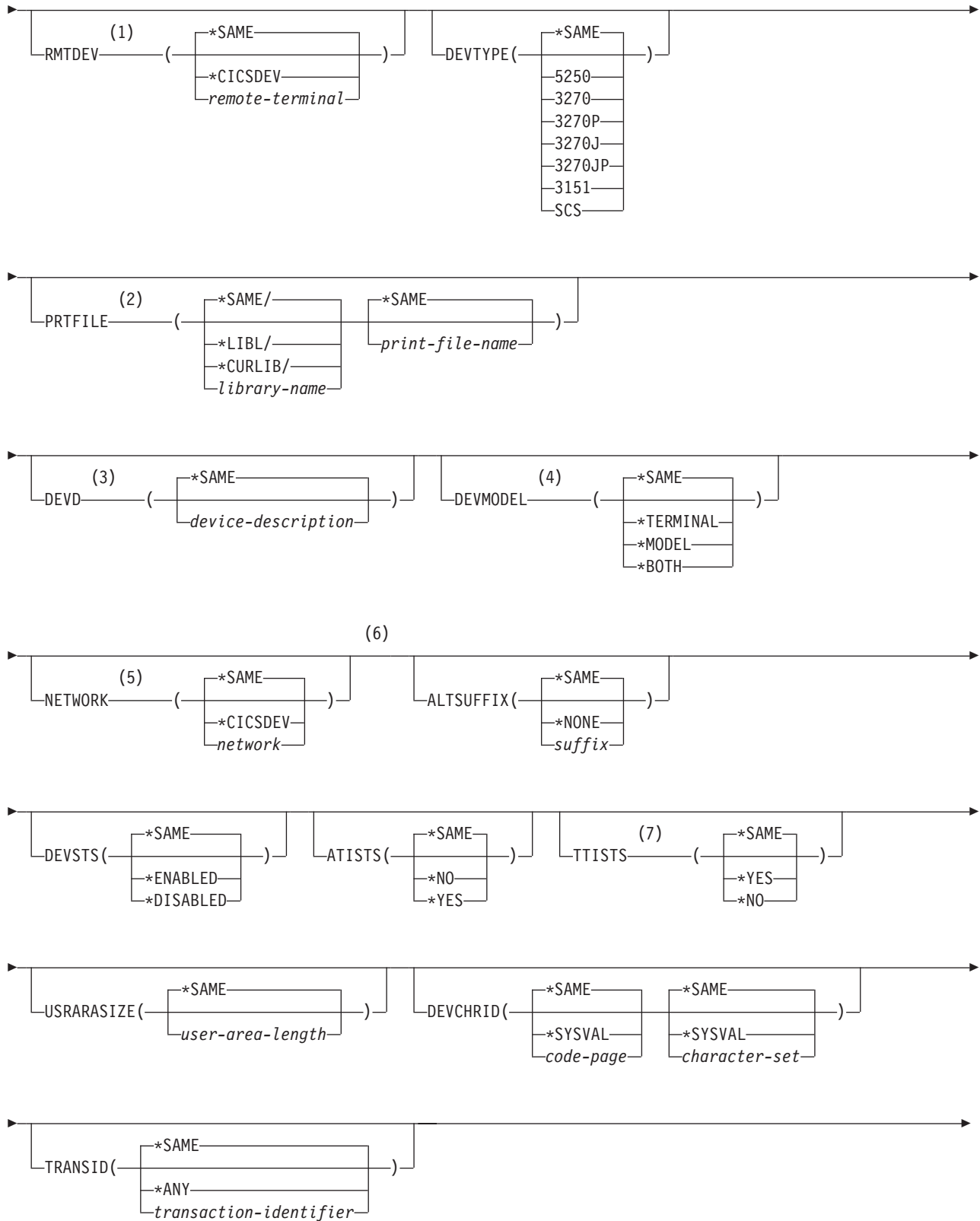
This command adds a TCT entry called AP01 to group ACCT in OS/400 library QCICSSAMP. The device type is 5250 (by default). The definition may be used both as a model for 5250 device types and as a terminal definition for OS/400 device DEVICE1.

Additional examples of ADDCICSTCT commands are given in “Example TCT definitions” on page 100.

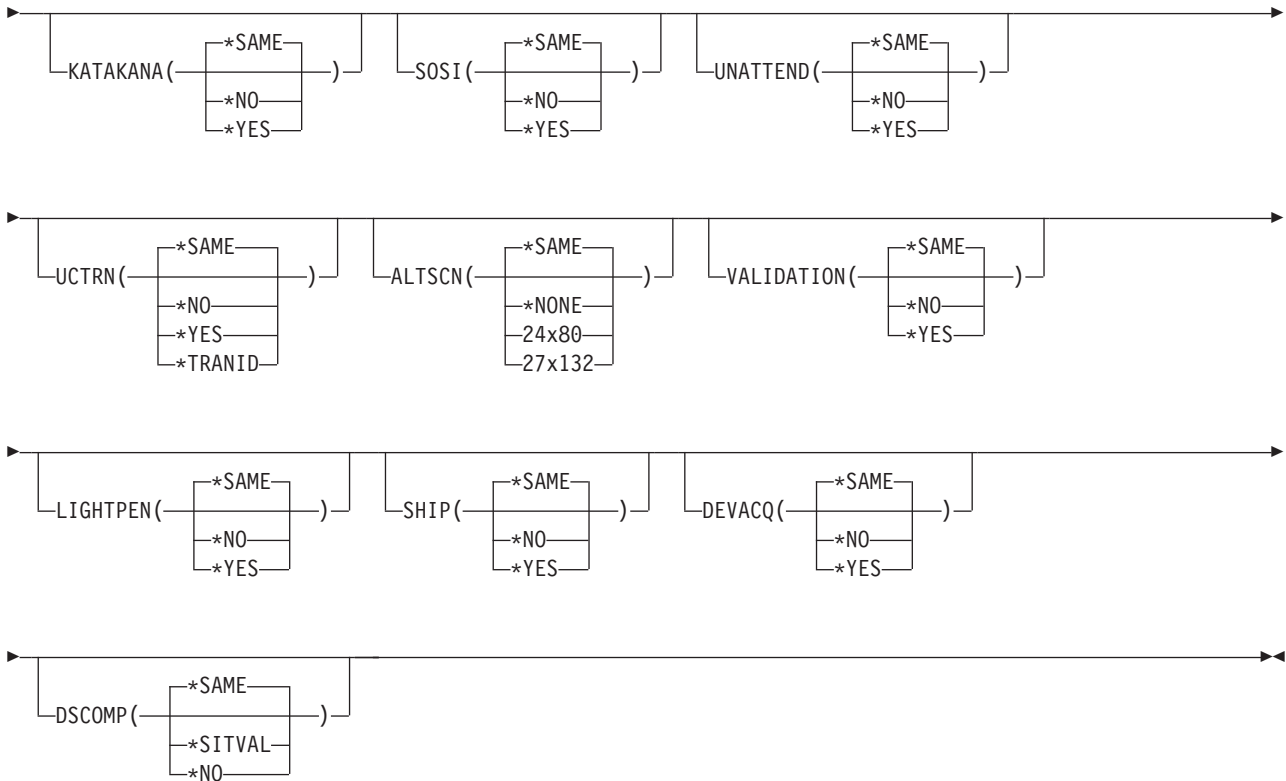
Using the CHGCICSTCT command

Job: B,I Pgm: B,I REXX: B,I Exec





CHGCICSTCT



Notes:

- 1 The RMTDEV parameter is not valid when SYSID(*NONE) is specified.
- 2 The PRTFILE parameter is valid, and required, when DEVTYPE(SCS) is specified.
- 3 The DEVD parameter is valid, and required, when SYSID(*NONE) is specified, DEVTYPE is not SCS, and DEVMODEL is not *MODEL.
- 4 The DEVMODEL parameter is valid only when DEVTYPE(5250), DEVTYPE(3270), DEVTYPE(3270J), or DEVTYPE(3151) is specified.
- 5 The NETWORK parameter is not valid when DEVMODEL(*MODEL) is specified.
- 6 All parameters preceding this point can be specified positionally.
- 7 TTISTS(*NO) is valid only when ATISTS(*YES) is specified.

Function

Use the Change CICS/400 Terminal Control Table (CHGCICSTCT) command to change an entry in the TCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter name of the group containing the TCT entry is to be changed.

group-name: The group name may have a maximum length of 10 characters.

The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS device (CICSDEV)

The terminal id that will be used to refer to this resource. If this TCT entry is defining a model terminal, this parameter identifies the entry, rather than a specific terminal.

terminal-identifier: The terminal identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Optional parameters**CICS system (SYSID)**

The system identifier defined in the TCS Table of the remote device. This parameter is required only for remote terminals. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***SAME:** The value currently specified in the TCT entry will remain the same.

***NONE:** The terminal is defined to the same control region in which it is being used.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote CICS device (RMTDEV)

The identifier by which the terminal is known on the remote system. This parameter is not valid when SYSID(*NONE) is specified.

Possible values are:

***SAME:** The value currently specified in the TCT entry will remain the same.

***CICSDEV:** The terminal associated with the TCT is to be used.

remote-terminal: The remote terminal may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Device type (DEVTYPE)

Indicates the type of terminal.

Possible values are:

***SAME:**

The value currently specified in the TCT entry will remain the same.

5250: Terminal supporting 5250 data stream.

3270: Terminal supporting 3270 data stream.

3270P: Printer supporting 3270 data stream.

3270J: Double-byte capable display.

3270JP: Double-byte capable printer.

3151: ASCII display.

SCS: Printer supporting SCS data stream.

Print file (PRTFILE)

The name of the print file to be used by the terminal. This is only valid when DEVTYPE(SCS) is specified.

Possible library values are:

***SAME:** The value currently specified in the TCT entry will remain the same.

***LIBL:** The library list for the job that is associated to the control region is used to locate the file.

***CURLIB:** The current library for the job that is associated to the control region is used to locate the file. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

Possible file name values are:

***SAME:** The value currently specified in the TCT entry will remain the same.

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

Device description (DEVVD)

The OS/400 device name that is associated with the terminal. This is only valid when SYSID(*NONE) is specified, DEVTYPE is not SCS, and DEVMODEL is not *MODEL.

Possible values are:

***SAME:** The value currently specified in the TCT entry will remain the same.

device-description: The device description may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Autoinstall model (DEVMODEL)

Indicates whether or not this terminal can be used as a model to autoinstall further terminals. This is only valid when DEVTYPE(5250), DEVTYPE(3270), DEVTYPE(3270J), or DEVTYPE(3151) is specified.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***TERMINAL:** The terminal cannot be used as a model to autoinstall further terminals.

***MODEL:** The terminal can only be used as a model to autoinstall further terminals.

***BOTH:** The terminal can be used as a model to autoinstall further terminals.

Network (NETWORK)

The symbolic network name used to identify the logical unit as it is known

throughout the network. The name is supplied to VTAM system definition and is used to build the node initialization block (NIB). This is not valid when DEVMODEL(*MODEL) is specified.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***CICSDEV:** The terminal suffixed with 4 blanks will be used as the symbolic network name.

network: The network may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Alternate suffix (ALTSUFFIX)

Enter a number to specify an alternate suffix for this device.

Possible values are:

***SAME:** The value currently defined in the TCT will remain the same.

***NONE:** The device will not have a user defined alternate suffix.

suffix: A number in the range 0 through 9. This numeric suffix will be appended by BMS to map set names, if the transaction uses the alternate screen size (or if the default and alternate screen sizes are the same).

Status (DEVSTS)

Indicates whether or not the terminal can be used by the control region when the TCT entry is defined to the runtime resource table definition Terminal Control facility.

Possible values are:

***SAME:** Keep the current value specified in the TCT entry.

***ENABLED:** The terminal can be used by the control region.

***DISABLED:** The terminal cannot be used by the control region.

ATI supported (ATISTS)

Indicates whether the terminal can be used by transactions that are initiated automatically by the control region, or in an ISC session. An ISC session is for transactions that are using the terminal as an alternate facility to communicate with another CICS system.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO:** The terminal cannot be used by the control region ATI facility.

***YES:** The terminal can be used by the control region ATI facility.

Transaction entry supported (TTISTS)

Indicates whether transactions may be started at this terminal.

Note: When ATISTS(*NO) is specified, this parameter must be *YES.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***YES:** Transactions may be started at this terminal.

***NO:** Transactions may not be started at this terminal. Transactions running on this terminal must be started by ATI or by EXEC CICS START commands.

User area size (User area size (USRARASIZE))

The length of the user area associated with the terminal.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

user-area-length: The user-area-length must be numeric, greater than or equal to 0 and less than or equal to 255.

Character identifier (DEVCHRID)

The code page and character set to be used with the terminal.

Possible values are:

Element 1: Code page

The code page to be used.

***SAME:** Keep the value currently specified in the TCT entry.

***SYSVAL:** Use the OS/400 system code page.

code-page: The code-page-number must be numeric, greater than or equal to 1 and less than or equal to 65 535.

Element 2: Graphic character set

The graphic character set to be used.

***SAME:** Keep the value currently specified in the TCT entry.

***SYSVAL:** Use the OS/400 system character set.

character-set: The character-set must be numeric, greater than or equal to 1 and less than or equal to 65 535.

Transaction (TRANSID)

The transaction identifier defined in the PCT, when it is the only transaction allowed to run on the terminal.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***ANY:** Any transaction can be executed on the terminal.

transaction-identifier: The transaction identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Japanese alphabet supported (KATAKANA)

Indicates whether Katakana support is required. Katakana terminals cannot display mixed case output; uppercase characters appear as uppercase English characters, but lowercase characters appear as Katakana characters. BMS provides automatic uppercase translation of lowercase Latin characters, in BMS fields where the CASE=MIXED option is specified, if Katakana support is required on the terminal where the map is to be displayed.

User written application may query this terminal attribute and provide additional support if required. Possible values are:

***SAME:** Keep the value currently specified in the TCT entry. ***NO:** The terminal does not require Katakana support.

***YES:** The terminal does require Katakana support.

User specified DBCS data (SOSI)

Indicates whether or not the terminal has mixed EBCDIC/DBCS field capability.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO** The terminal does not have mixed EBCDIC/DBCS field capability.

***YES:** The terminal has mixed EBCDIC/DBCS field capability.

Unattended mode (UNATTEND)

Indicates whether or not the mode of operation for the terminal is unattended.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO:** The terminal is to have an attended mode of operation.

***YES:** The terminal is to have an unattended mode of operation.

Auto uppercase translation (UCTRN)

Indicates whether or not the terminal is to have uppercase translation done by CICS/400.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO:** CICS/400 translates lowercase characters in data input from this terminal to uppercase only if the transaction requires data in uppercase (UCTRN option in ADDCICSPCT command). CICS/400 never translates lowercase characters in a transaction ID.

***YES:** CICS/400 always translates lowercase characters in input (data or transaction ID) from this terminal to uppercase.

***TRANID:** CICS/400 translates lowercase characters in data input from this terminal to uppercase only if the transaction requires data in uppercase (UCTRN option in ADDCICSPCT command). CICS/400 always translates a transaction ID that contains any lowercase characters to be completely uppercase.

Table 24 on page 274 shows the results of the various combinations of UCTRN values in the TCT and PCT.

Alternate screen size (ALTSCN)

Indicates the alternate screen height and width associated with the terminal.

Possible values are:

***SAME:** Keep the value currently specified in the TCT table.

***NONE:** No alternate screen size will be used.

24x80: 24 rows and 80 columns will be used as the alternate screen size.

27x132: 27 rows and 132 columns will be used as the alternate screen size.

Validation capability (VALIDATION)

Indicates whether or not the terminal has validation capability. This consists of the mandatory fill and mandatory enter.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO:** The terminal does not have validation capability.

***YES:** The terminal has validation capability.

Light pen supported (LIGHTPEN)

Indicates whether or not the terminal has light pen capability.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO:** The terminal does not have light pen capability.

***YES:** The terminal has light pen capability.

CHGCICSTCT

Ship to another CICS system (SHIP)

Indicates whether or not the terminal definition is to be shipped to the other CICS system.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO:** The terminal definition will not be shipped to the other CICS system.

***YES:** The terminal definition will be shipped to the other CICS system.

ATI acquire (DEVACQ)

Indicates whether or not the terminal is to be acquired by transactions that are initiated automatically by the control region.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***NO:** The control region ATI facility will wait to initiate the transaction, until the terminal is in an acquire state.

***YES:** The control region ATI facility will acquire the terminal if it is not already in an acquire state, before initiating the transaction.

Datastream compression (DSCOMP)

Indicates whether or not a datastream sent to this terminal is to be compressed.

Possible values are:

***SAME:** Keep the value currently specified in the TCT entry.

***SITVAL:** Data output to this terminal is compressed if the DSCOMP option in the SIT (system initialization table) specifies the type of this terminal (3270 or 5250) or *ALL.

***NO:** Data output to this terminal is never compressed.

Table 25 on page 275 shows the results of the various combinations of DSCOMP values in the TCT and SIT.

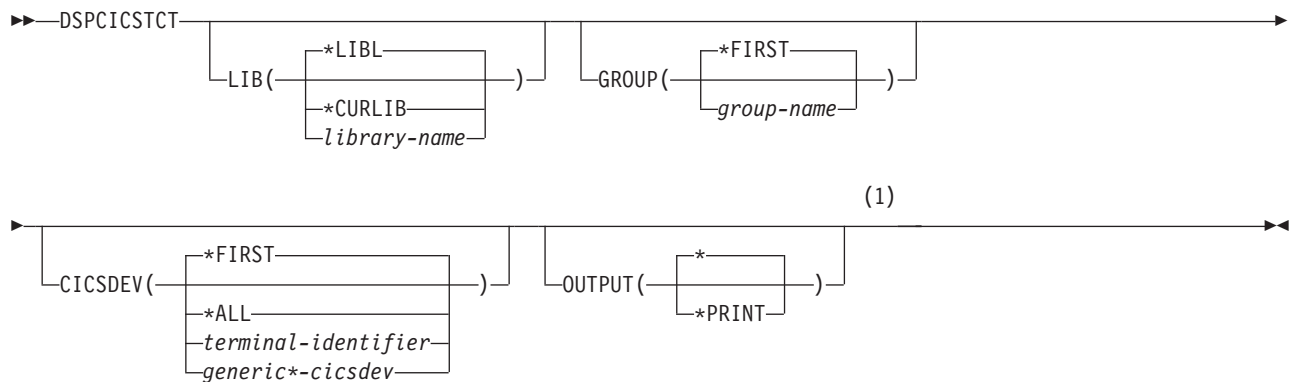
Examples

```
CHGCICSTCT LIB(QCICSSAMP) GROUP(ACCT)
          CICSDEV(AP01) SHIP(*YES)
```

This command changes the TCT entry called AP01 in group ACCT in OS/400 library QCICSSAMP. The definition may now be shipped to other CICS systems.

Using the DSPCICSTCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Terminal Control Table (DSPCICSTCT) command to display a TCT entry. You can only view this entry; you can neither make changes to it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group. Possible values are:

- *LIBL:** The library list is used to locate the first OS/400 library that contains the group.
- *CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.
- library-name:* The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the TCT entry to be displayed.

Possible values are:

- *FIRST:** No group is specified, the first group found is used.
- group-name:* The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS device (CICSDEV)

The name of the TCT entry to be displayed. This is also known as the terminal used to work with a device.

Possible values are:

- *FIRST:** Display the first TCT entry.
- *ALL:** Display all of the TCT entries.

DSPCICSTCT

terminal-identifier: The terminal identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-cicsdev*: Specify the generic name of the terminal. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all entries with terminal identifiers beginning with the generic name are displayed. If an asterisk is not included with the generic name, the system assumes the value to be the complete terminal.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT**: The output is printed with the job spool output.

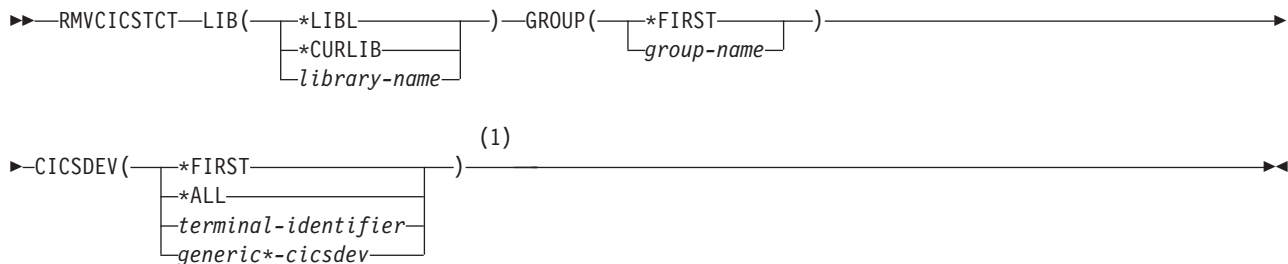
Examples

```
DSPCICSTCT LIB(QCICSSAMP) GROUP(ACCT) CICSDEV(*ALL)
```

This command displays all TCT entries located in group ACCT in OS/400 library QCICSSAMP.

Using the RMVICSTCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Terminal Control Table (RMVICSTCT) command to delete an entry from the TCT.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL**: The library list is used to locate the first OS/400 library that contains the group.

***CURLIB**: The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the TCT entry to be removed.

Possible values are:

***FIRST**: No group is specified, the first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS device (CICSDEV)

The name of the TCT entry to be removed. This is known as the terminal used to work with a device.

Possible values are:

***FIRST**: Remove the first TCT entry.

***ALL**: Remove all of the TCT entries.

terminal-identifier: The terminal identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-cicsdev*: Specify the generic name of the terminal. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all entries with terminal identifiers beginning with the generic name are removed. If an asterisk is not included with the generic name, the system assumes the value to be the complete terminal.

Examples

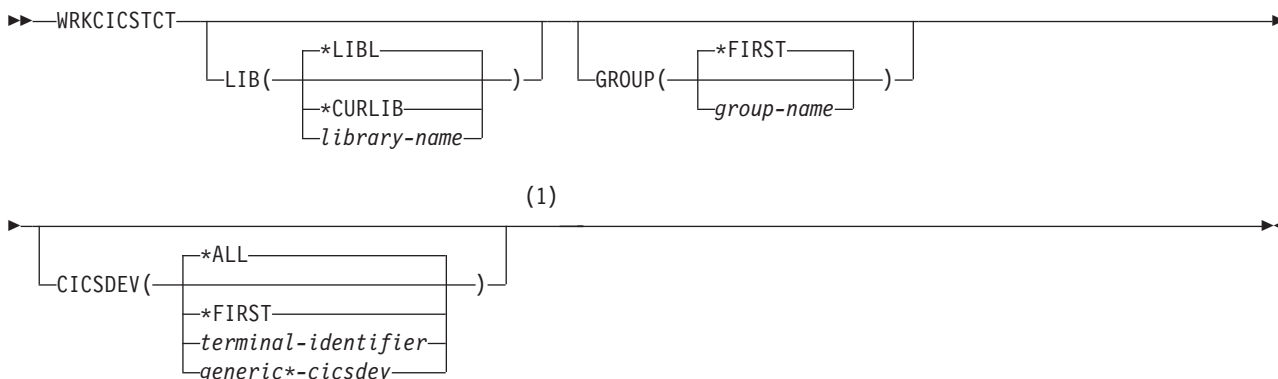
```
RMVICSTCT LIB(QCICSSAMP) GROUP(ACCT)
          CICSDEV(ABC*)
```

This command removes all TCT entries, that start with ABC and end with anything, located in group ACCT in OS/400 library QCICSSAMP.

WRKCICSTCT

Using the WRKCICSTCT command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Terminal Control Table (WRKCICSTCT) command to list entries in the TCT. You can change, remove, copy or display entries, or add new entries to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the TCT entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

CICS device (CICSDEV)

Enter the name of the TCT entry to be listed. This is the name of the terminal.

Possible values are:

***ALL:** List all TCT entries.

***FIRST:** List the first TCT entry.

terminal-identifier: The terminal identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

generic-cicsdev*: Specify a generic terminal identifier. A generic name is a string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all entries with terminal identifiers beginning with the generic name are listed. If an asterisk is not included with the generic name, the system assumes the value to be the complete terminal.

Examples

```
WRKICSTCT LIB(QCICSSAMP) GROUP(ACCT)
```

This command lists all TCT entries located in group ACCT in OS/400 library QCICSSAMP.

Managing temporary storage resource definitions

A temporary storage queue is a file used by an application program to store data for later retrieval. A temporary storage table (TST) entry needs to be created for all recoverable and all remote temporary storage queues. You do not need to create TST entries for local, unrecoverable temporary storage queues.

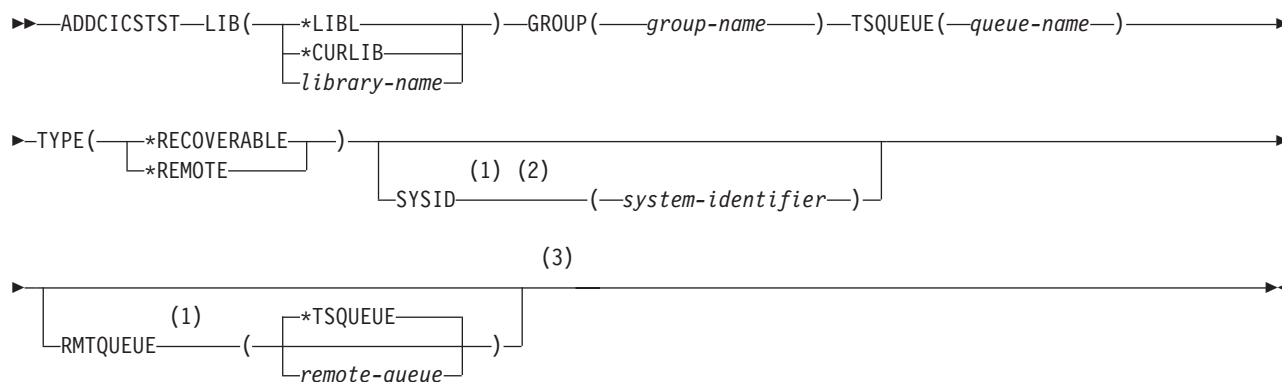
You can create a TST entry with a generic name that can be used as a prefix to the full name that will be supplied by the application programs in EXEC CICS temporary storage commands. A generic name enables you to create one entry to apply to many temporary storage queues. CICS/400 will use the TST entry associated with the generic name. You must take care, therefore, that there is no ambiguity in your TS queue names.

See “Defining temporary storage and transient data files” on page 63 and “Temporary storage considerations” on page 84 for guidance information.

ADDCICSTST

Using the ADDCICSTST command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The SYSID parameter and the RMTQUEUE parameter are valid only when TYPE(*REMOTE) is specified.
- 2 The SYSID parameter is required when TYPE(*REMOTE) is specified.
- 3 All parameters preceding this point can be specified positionally.

Function

Use the Add CICS/400 Temporary Storage Table (ADDCICSTST) command to add an entry to the TST.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the group.

***CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group to which this TST entry is to be added.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Queue (TSQUEUE)

Enter either an eight-character name of a queue or a partial, generic name. This is the name that will be used in EXEC CICS temporary storage commands. This name is also used to identify this TST entry.

queue-name: The queue name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #. If a partial name is specified, it will be treated as a generic name and all TST entries beginning with the generic name will be eligible for processing.

Type (TYPE)

Indicates whether the temporary storage queue is recoverable or remote.

Possible values are:

***RECOVERABLE:** The temporary storage queue is recoverable.

***REMOTE:** The temporary storage queue is remote.

Optional parameters**CICS system (SYSID)**

Enter the identifier of the system that owns the temporary storage queue. The system should have a TCS entry. This parameter is required only for remote queues, that is when the **Type (TYPE)** parameter contains *REMOTE. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote queue name (RMTQUEUE)

Enter the name by which the temporary storage queue is known on the remote system. This parameter is valid only for remote queues, that is when the **Type (TYPE)** parameter contains *REMOTE.

Possible values are:

***TSQUEUE:** The local and remote temporary storage queue names are the same. The name in the TSQUEUE parameter will be used.

remote-queue: The temporary storage queue name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Examples

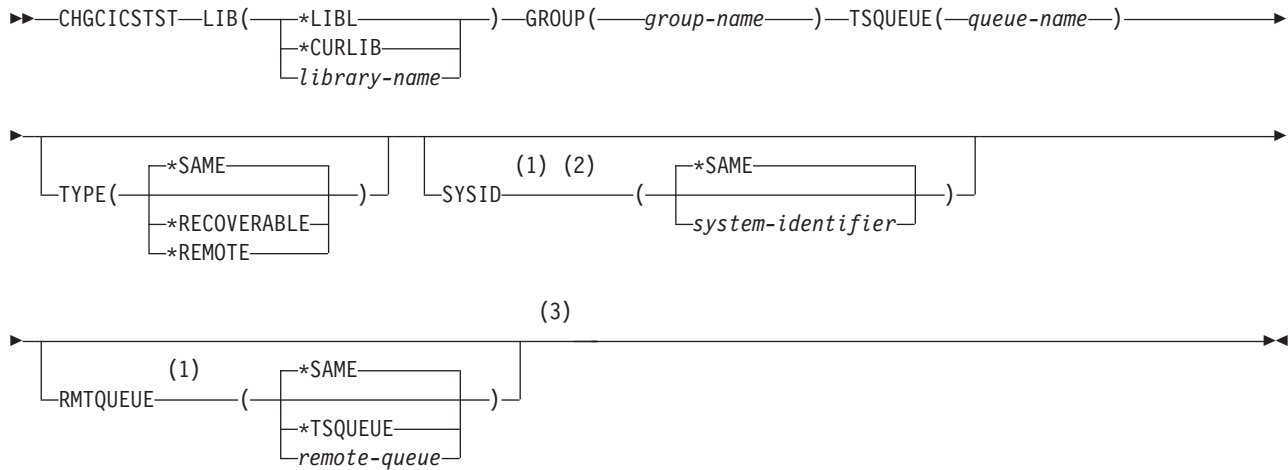
```
ADDCICSTST LIB(CICSWORK) GROUP(ACCT)
          TSQUEUE(AU) TYPE(*RECOVERABLE)
```

This command adds a TST entry called AU to group ACCT in OS/400 library CICSWORK, as a recoverable queue type. This definition will apply to all temporary storage queues with names beginning with AU.

CHGCICSTST

Using the CHGCICSTST command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 The SYSID parameter and the RMTQUEUE parameter are valid only when TYPE(*REMOTE) is specified.
- 2 The SYSID parameter is required when TYPE(*REMOTE) is specified.
- 3 All parameters preceding this point can be specified positionally.

Function

Use the Change CICS/400 Temporary Storage Table (CHGCICSTST) command to change an entry in the TST.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.
Possible values are:

- *LIBL:** The library list is used to locate the first OS/400 library that contains the group.
- *CURLIB:** The current library contains the group. If no current library is specified, the QGPL library is used.
- library-name:* The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group that contains the TST entry to be changed.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Queue (TSQUEUE)

Enter either an eight-character name of a queue or a partial, generic name. This is the name that will be used in EXEC CICS temporary storage commands. This name is also used to identify this TST entry.

queue-name: The queue name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #. If a partial name

is specified, it will be treated as a generic name and all TST entries beginning with the generic name will be eligible for processing.

Optional parameters

Type (TYPE)

Indicates whether the temporary storage queue is recoverable or remote.

Possible values are:

***SAME:** Keep the value currently specified in the TST entry.

***RECOVERABLE:** The temporary storage queue is recoverable.

***REMOTE:** The temporary storage queue is remote.

CICS system (SYSID)

Enter the identifier of the system that owns the temporary storage queue. The system should have a TCS entry. This parameter is required only for remote queues, that is when the **Type (TYPE)** parameter contains *REMOTE. If this parameter contains the ID of the local system, the entry will be treated as if *NONE has been entered.

Possible values are:

***SAME:** Keep the value currently specified in the TST entry.

system-identifier: The system identifier may have a maximum length of 4 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Remote queue name (RMTQUEUE)

Enter the name by which the temporary storage queue is known on the remote system. This parameter is valid only for remote queues, that is when the **Type (TYPE)** parameter contains *REMOTE.

Possible values are:

***SAME:** Keep the value currently specified in the TST entry.

***TSQUEUE:** The local and remote temporary storage queue names are the same. The name in the TSQUEUE parameter will be used.

remote-queue: The temporary storage queue name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Examples

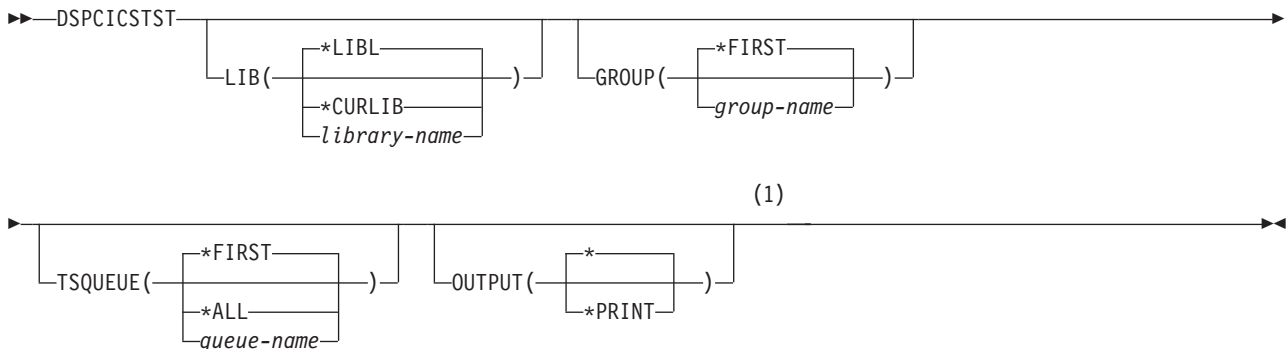
```
ADDCICSTST LIB(CICSWORK) GROUP(ACCT)
  TSQUEUE(AU) TYPE(*REMOTE) SYSID(SYS1)
```

This command changes the TST entry called AU in group ACCT in OS/400 library CICSWORK. The queue has been redefined as a remote queue on system SYS1.

DSPCICSTST

Using the DSPCICSTST command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Display CICS/400 Temporary Storage Table (DSPCICSTST) command to display a TST entry. You can only view this entry; you can neither make changes to it nor delete it.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the TST entry to be displayed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Queue (TSQUEUE)

Enter the name of the TST entry to be displayed. This could be either a complete name or a generic name.

Possible values are:

***FIRST:** Display the first TST entry.

***ALL:** Display all TST entries.

queue-name: The queue name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #. If a

partial name is specified, it will be treated as a generic name and all TST entries beginning with the generic name will be eligible for processing.

Location of output (OUTPUT)

Enter the location of the output from this command.

Possible values are:

*: The output is either displayed (if requested by an interactive job) or printed with the job spool output (if requested by a batch job).

***PRINT**: The output is printed with the job spool output.

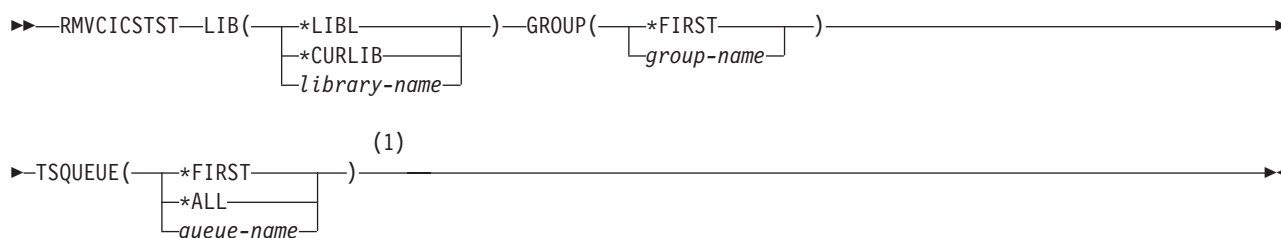
Examples

```
DSPCICSTST LIB(CICSWORK) GROUP(ACCT) TSQUEUE(*ALL)
```

This command displays all TST entries located in group ACCT in OS/400 library CICSWORK.

Using the RMVICSTST command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Remove CICS/400 Temporary Storage Table (RMVICSTST) command to delete an entry from the TST.

Required parameters

Library (LIB)

Enter the name of the OS/400 library that contains the CICS/400 group.

Possible values are:

***LIBL**: The library list is used to locate the first OS/400 library that contains the group.

***CURLIB**: The current library contains the group. If no current library is specified, the QGPL library is used.

library-name: The name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the TST entry to be removed.

Possible values are:

***FIRST**: No group is specified. The first group found is used.

RMVICICSTST

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Queue (TSQUEUE)

Enter the name of the TST entry to be removed. This could be either a complete or a generic name.

Possible values are:

***FIRST**: Remove the first TST entry.

***ALL**: Remove all TST entries.

queue-name: The queue name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #. If a partial name is specified, it will be treated as a generic name and all TST entries beginning with the generic name will be eligible for processing.

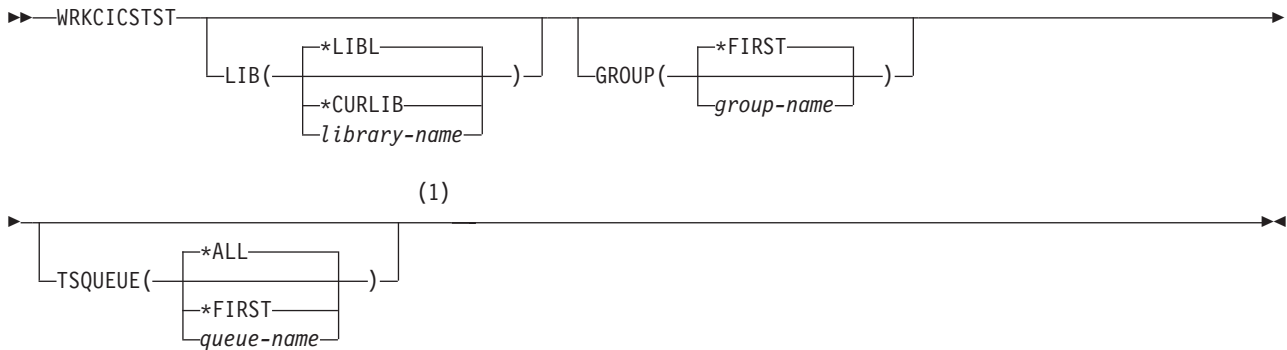
Examples

```
RMVICICSTST LIB(CICSWORK) GROUP(ACCT)
             TSQUEUE(ABC*)
```

This command removes all TST entries, that start with ABC and end with anything, located in group ACCT in OS/400 library CICSWORK.

Using the WRKICICSTST command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

- 1 All parameters preceding this point can be specified positionally.

Function

Use the Work with CICS/400 Temporary Storage Table (WRKICICSTST) command to list entries in the TST. You can change, remove, copy and display entries, or add new entries to the list.

Optional parameters

Library (LIB)

Enter the name of the OS/400 library that contains the group.

Possible values are:

***LIBL:** The library list is used to locate the first OS/400 library that contains the CICS/400 group.

***CURLIB:** The current library contains the CICS/400 group. If no library is specified as the current library, the QGPL library is used.

library-name: Specify the name of the OS/400 library that contains the group.

Group (GROUP)

Enter the name of the group containing the TST entries to be listed.

Possible values are:

***FIRST:** No group is specified. The first group found is used.

group-name: The group name may have a maximum length of 10 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #.

Queue (TSQUEUE)

Enter the name of the TST entry to be listed. This is the name that is used in temporary storage EXEC CICS commands.

Possible values are:

***ALL:** List all TST entries.

***FIRST:** List the first TST entry.

queue-name: The queue name may have a maximum length of 8 characters. The first character must be alphabetic, or one of the special characters, \$, @, or #. The remaining characters can be alphanumeric or \$, @, or #. If a partial name is specified, it will be treated as a generic name and all TST entries beginning with the generic name will be eligible for processing.

Examples

```
WRKICSTST LIB(CICSWORK) GROUP(ACCT)
```

This command lists all TST entries located in group ACCT in OS/400 library CICSWORK.

WRKICSTST

Chapter 11. Autoinstall for terminal definitions

General-use programming interface

This chapter describes autoinstallation of terminal resource definitions.

- “How autoinstall terminal identifiers are generated” on page 307
- “Defining an autoinstall model terminal using masking” on page 308
- “Using an autoinstall control program” on page 309
- “Writing your own autoinstall program” on page 312
 - “Autoinstalling terminal definitions” on page 313
 - “Deleting autoinstalled terminal definitions” on page 317

CICS needs a TCT definition for each terminal device with which it communicates. However, a CICS/400 system may have many connected terminals, and there is a great deal of keying work associated with creating individual TCT entries for each terminal. Because many of your terminals have identical properties, you can use the autoinstall facility to reduce the number of TCT definitions you have to create. Instead of a TCT entry for each terminal, you create a model TCT definition that could apply to many terminals of the same type. When you log on from a terminal, CICS/400 uses the model to create dynamically an entry for that terminal in the TCT. The terminal identifier is generated by CICS/400 from the OS/400 device description information contained in the SIT. This process is referred to as *terminal masking*. See the DEVCTL parameter of the SIT in “Using the ADDCICSSIT command” on page 239.

Optionally, you can specify a CICS program to be called by CICS/400 to generate a terminal identifier. This program is referred to as an autoinstall control program. CICS/400 calls the program referenced by the AEGTCACP PPT entry defined to the control region, after the terminal masking process has completed.

With autoinstall, the amount of storage required for terminal definitions is reduced. Definitions are installed when they are needed, that is when the user logs on from the terminal. The autoinstallation process is summarized in Figure 58 on page 307.

Definitions are deleted when the user has logged off, and the autoinstall inactivity limit has expired. The autoinstall inactivity limit is specified in the DEVCTL parameter of the SIT. See “Managing system initialization resource definitions” on page 239.

Autoinstall models may be created for 3270 displays, including those with the double-byte capability, 5250 displays, and ASCII terminals. The DEVMODEL parameter of the TCT is used to indicate whether the definition can be used as a model terminal resource definition. See “Using the ADDCICSTCT command” on page 275.

Client terminals are a special case of autoinstalled terminals. While able to take advantage of autoinstall model terminal definitions, they also possess some unique characteristics.

- Client terminal installation is driven by the internal transaction CTIN. This is invoked by CICS client external presentation interface (EPI) applications to install a remote-terminal entry in the CICS control region.

- The client-terminal identifier is not defined using the SIT mask and AEGTCACP exit program described in this chapter. It is generated automatically by the CTIN transaction.
- Client terminals are deleted either by a CTIN uninstall request issued by the client EPI application, or by a CCIN install request for a client that is already installed.

The CICS Client INstall transaction (CCIN) and CICS Terminal INstall transaction (CTIN) can be found in group AEGCLI.

For a full description of client terminals, refer to *CICS for iSeries Intercommunication*.

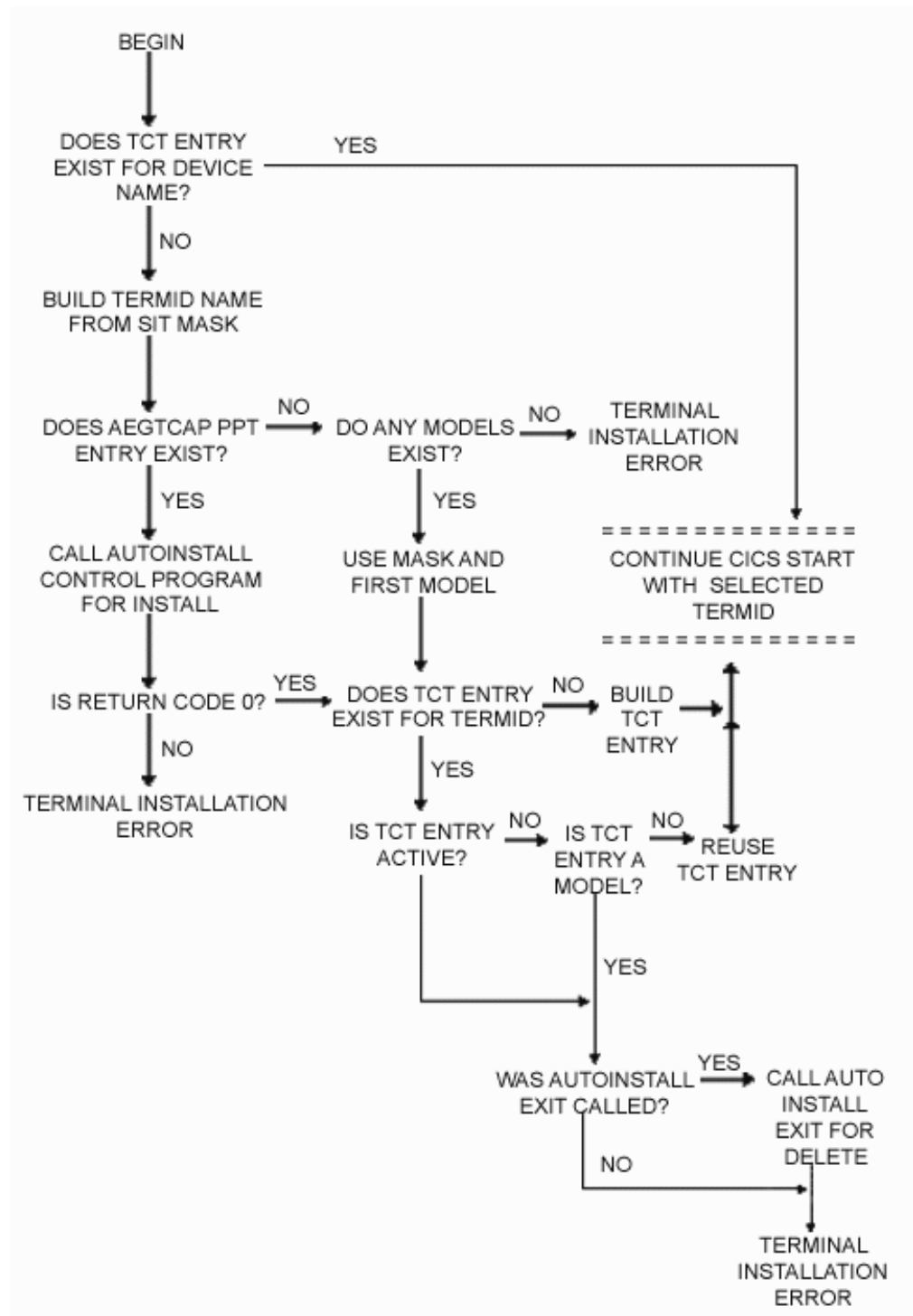


Figure 58. Flowchart of the autoinstall functions

How autoinstall terminal identifiers are generated

CICS/400 offers two ways of generating autoinstall terminal resource definitions. If, when the user attempts to log on at the terminal, no TCT definition is found for the terminal, one of the following may happen:

- If an AEGTCACP PPT entry is not defined to the control region, CICS/400 uses the masking facilities defined in the SIT to generate a terminal identifier.
- If an AEGTCACP PPT entry is properly defined to the control region, CICS/400 first uses the masking facilities to generate a potential terminal identifier. CICS/400 then calls the autoinstall control program referenced by the AEGTCACP PPT entry. Assuming a successful return from the autoinstall control program, CICS/400 uses the terminal identifier returned from that program.

The use of an autoinstall control program is optional; by default, masking is used to autoinstall terminals. If you wish to use an autoinstall control program, you must supply an AEGTCACP PPT entry for it.

Defining an autoinstall model terminal using masking

The DEVCTL parameter in the SIT is used to define an autoinstall mask. See “Using the ADDCICSSIT command” on page 239 for details of this parameter. The mask determines which characters of the OS/400 device description CICS is to use to generate a CICS terminal identifier (termid) when a STRCICSUSR command is entered from an iSeries device.

You can define TCT entries as autoinstall model entries for the various device types attached to your CICS system.

When a STRCICSUSR command is entered from a terminal, CICS searches the TCT for a match on the OS/400 device name. If no match is found, CICS generates a termid from the OS/400 device name, based on the autoinstall mask in the SIT. Information about OS/400 device names can be found in the File Management topic in the iSeries Information Center.

Using the generated termid, CICS searches the TCT entries to determine whether there is an autoinstall model defined.

- If there is no autoinstall model defined, the STRCICSUSR command is terminated with an error indicating that autoinstall has failed.
- If there is an autoinstall model defined, then CICS/400 uses the first model found.

The generated termid is used in a search of the TCT again to ensure that the termid is not already active within the CICS system. If the terminal table entry exists and is active, the STRCICSUSR command is terminated with an autoinstall failure.

If the terminal table entry exists and is a model, the STRCICSUSR command is terminated with an autoinstall failure.

If there is an existing entry that is not active, it is made active. If there are no TCT entries, the terminal control table entry for the autoinstall termid is created, using the model, the generated terminal identifier, and information from OS/400 about the device. The user is allowed into the CICS system.

It is important that the autoinstall mask used within the SIT results in unique terminal identifiers (termids) being generated by CICS. If the generated termids are not unique, CICS users may not be able to get access to the system, because CICS may interpret the termid as already installed and in use.

Examples of autoinstall masks and termids generated by selecting specified positions of the device name are shown in Table 26 on page 309.

Table 26. Examples of autoinstall masks

Mask	Device name	Terminal ID
0245	PWS002S1	PS02
0234	DSP01	DP01
0234	DSP02	DP02
0765	PWS002S1	P1S2
0A7B	PWS002S2	PA2B
0A7B	PWS002S1	PA1B

Using an autoinstall control program

CICS/400 supplies source for a sample autoinstall control program. Alternatively, you can create your own customized autoinstall control program. You can use the supplied program source and customize it to your own needs, or you can write the program yourself. It is highly recommended that you customize the supplied program source, because it contains the proper structures for passing parameters between CICS/400 and your program.

The CICS/400-supplied autoinstall control program

The CICS/400-supplied autoinstall control program is a COBOL/400 program, named AEGTCACP. Like the installation verification program source, the source code for AEGTCACP, and for its associated copybook DFHTCUDO, are supplied as members of the source file QCICSSAMP/QLBLSRC. See Chapter 3, “Installing CICS/400” on page 21 for information on installing the sample code.

The default action of the autoinstalled terminal supplied program AEGTCACP on installation is to select the first model in the list, move the terminal identifier generated by masking to the output terminal identifier, set the return code, and return to CICS/400. If there are no models in the list, it returns with no action.

The default action on autoinstalled terminal deletion is to address the passed parameter list, and return to CICS/400 with no action.

How does CICS/400 know to call an autoinstall control program?

If an AEGTCACP PPT entry is defined to the CICS/400 system, the program referred to by that entry is called during installation of an autoinstall terminal and when an autoinstalled terminal is being deleted. To use an autoinstall control program, you should:

- Create a PPT definition for AEGTCACP. The PGMOBJ parameter in the PPT definition must refer to the program object of the autoinstall control program you intend to use. Note that the program object can be given any name you wish.
- Translate and compile the autoinstall control program. Be sure you create the program object with the proper name.

Creating a PPT definition for AEGTCACP

A PPT entry must be created with PGMID parameter of AEGTCACP. For example,

```
ADDICSPPT LIB(QTEST) GROUP(AUTO) PGMID(AEGTCACP)
          SYSID(*NONE) PGMOBJ(CICSWORK/AEGTCACP) CICSDEBUG(*NODEBUG)
```

The PPT definition must be defined as local (SYSID(*NONE)) and must be enabled.

If the PPT definition is defined as remote, or is disabled, there will be no autoinstall processing. **Remember, if there is no PPT definition for AEGTCACP, CICS/400 will use masking to define autoinstall terminal identifiers.**

Translating the autoinstall control program

Whether or not you use the CICS/400-supplied program or your own customized autoinstall control program, the program must be translated and compiled using the CICS/400 supplied CL command CRTICSCBL. The following example shows the CRTICSCBL command for the CICS/400-supplied program AEGTCACP in source file QCICSSAMP/QLBLSRC. To translate the program, enter:

```
CRTICSCBL
```

and press F4. The panel in Figure 59 is displayed. You have to type in only the first three fields highlighted. The source member field should be entered if the source member name differs from the program object name. The remaining fields have acceptable defaults.

```
                Create CICS COBOL/400 (CRTICSCBL)
Type choices, press Enter.
Program . . . . . PGM          > AEGTCACP      1
Library . . . . .             > CICSWORK     2
Source File . . . . . SRCFILE  > QLBLSRC     3
Library . . . . .             > QCICSSAMP
Source Member . . . . . SRCMBR *PGM          4
Commitment control . . . . . COMMIT *CHG
Text description . . . . . TEXT    *SRCMBRTXT

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
```

Figure 59. The Create CICS COBOL screen

Notes:

- 1 This is the name of the OS/400 program object that is generated. The object name can be any name you wish, however the AEGTCACP PPT entry PGMOBJ parameter must refer to this object name.
- 2 This is the library that holds the OS/400 program object. This can be any existing library you wish.
- 3 This is the name of the source file that contains the program source code.

CICS/400 provides the source code for AEGTCACP in file QLBSRC library QCICSSAMP. If you have your own autoinstall program in another source file, you should enter that name here.

- 4 This is the name of the source member containing the source for your autoinstall control program.

For further details on the CRTICSCBL command, see the *CICS for iSeries Application Programming Guide*.

Security

Be sure that the users of your autoinstall control program are authorized to any objects referenced by your autoinstall control program; for example, writing records as part of autoinstall installation to an audit file. If your user is not authorized to write to that file, the program will get a NOTAUTH error on the write request. If the program does not allow for this possibility, the autoinstall control program fails and the user will not be allowed into the CICS system.

Program object security

You must ensure that the submitter of the STRCICS command for starting your CICS system is authorized to the PGMOBJ referred to by the AEGTCACP PPT definition. If the submitter is not authorized, an error message is written to the system message log CSMT and to the STRCICS process job log, indicating that autoinstall will not be active for that CICS system. For information about defining the CSMT log, see “Using the ADDCICSDCT command” on page 161.

In addition, each user of the CICS system must be authorized to the autoinstall control program. If an individual user who attempts to log into the CICS system is not authorized to the program, that user will be prevented from autoinstalling into that CICS system.

Autoinstall delete security considerations

When the autoinstall control program is called in an IC batch shell during CATD transaction processing, the user profile of the submitter of the STRCICS command is used as the user profile for the IC batch job.

However, if the autoinstall control program is called because autoinstall has failed within a shell, the program is called within the process where the STRCICSUSR command was issued. In this case, the user profile for that process is used.

Batch shell considerations

Autoinstall delete processing is normally executed using the CICS/400-supplied transaction CATD. The CATD transaction is executed in an Interval Control batch shell. If you are using an autoinstall control program, you may need to define more Interval Control batch shells to your CICS system. The ITVCTL parameter of the ADDCICSSIT command is used to define the maximum and minimum number of batch shells for the CICS system.

Storage considerations

CICS/400 allocates a large internal table to hold the areas required by the autoinstall control program AEGTCACP. If you are using an autoinstall control program, your CICS/400 shells require 5KB more space to allow for autoinstall processing requirements.

CEMT INQUIRE/SET PROGRAM and EXEC CICS INQUIRE/SET PROGRAM

PPT entries within CICS/400 that begin with AEG are not allowed to be DISABLED in an active CICS system. Because the PPT entry for the autoinstall control program must be called AEGTCACP, you are not able to DISABLE the program using CEMT or the EXEC CICS SET PROGRAM equivalent. However, you are allowed to use an updated version of your autoinstall control program by setting a new copy of the program into the CICS system by using the CEMT SET PROGRAM(AEGTCACP) NEWCOPY command or its equivalent EXEC CICS SET PROGRAM(AEGTCACP) NEWCOPY in an application program.

Turning off autoinstall in an active CICS system

There are many ways to accomplish this. One way would be to discard all the autoinstall models in the CICS system. Another way would be to install an AEGTCACP PPT entry with the program status PGMSTS(*DISABLED), using either the CEDA transaction or the INSCICSGRP CL command. In addition, if the submitter of the STRCICS command is not authorized to the program object referred to by the AEGTCACP PPT entry, autoinstall processing is not active for that CICS system.

Attempting to install an AEGTCACP PPT entry defined as remote

If you attempt to install an AEGTCACP PPT entry that is defined as remote, CICS/400 rejects the installation request with an error message written to the CSMT log and the job log indicating that the AEGTCACP PPT entry has been rejected.

Writing your own autoinstall program

This section tells you how to tailor the CICS-supplied autoinstall program to extend the terminal naming facilities provided as part of the autoinstall process. You might want to do this if the device masking facility is inadequate to cope with your terminal naming conventions, or you require control over the choice of model name. You can use the source code of the supplied program and the customization example in this section as a basis. Alternatively, you can write your own program, if the supplied program does not meet your needs. You are recommended to try first a program based on that supplied with CICS/400. You must write your main program in COBOL/400, but this program may call programs in other languages. You can give any name to your customized program.

- The source code for the CICS/400-supplied AEGTCACP program is provided in source file QICSSAMP/QLBLSRC.
- COBOL copybook DFHTCUDO is provided in the same source file as AEGTCACP. This copybook should be used by your autoinstall control program.
- You can trap an abend in your customized program by making the program issue an EXEC CICS HANDLE ABEND command.
- You must ensure that your customized program is defined as a local program in the PPT. You cannot run an autoinstall control program in a remote CICS system.
- You should use the CRTICSCBL command to translate and compile your autoinstall control program. You should replace the name of the supplied program AEGTCACP with that of your customized autoinstall program. See "Translating the autoinstall control program" on page 310.

In addition to managing your autoinstall resource definition, your autoinstall control program can be customized to perform other processes. Its access to the command-level interface is that of a normal, nonterminal user task. Some possible uses are listed on page 319.

The control program is invoked when:

- An autoinstall install request is being processed.
- An autoinstall delete request has just been completed.
- An autoinstall request has previously been accepted by the user program, but the subsequent install process has failed. In this case, the control program is invoked as if for a delete request.

On each invocation of the autoinstall control program, a parameter list is passed using a communication area (COMMAREA), describing the function being performed (install or delete), and providing data relevant to the particular event.

Autoinstalling terminal definitions

The autoinstall control program is invoked when the following conditions apply:

- An AEGTCACP PPT definition exists in the CICS system. This is the PPT definition for the autoinstall control program.
- There is no TCT entry match for the device trying to log into the CICS system.
- Autoinstall processing has been completed to a point where information (a terminal identifier and autoinstall model name) from the control program is required to proceed.

The COMMAREA for an install request

Input to the program is through a COMMAREA.

Data area	Standard header
Byte 1	Request type (X'F0' for install)
Bytes 2-3	Component id ('TC')
Bytes 4-16	Reserved
Pointer areas	
Pointer 1	NETNAME_FIELD
Pointer 2	MODELNAME_LIST
Pointer 3	SELECTED_PARMS (return parameter data)
Pointer 4	DEVICE_FIELD

Figure 60. Autoinstall control program's COMMAREA for install

The COMMAREA, the layout of which is shown in Figure 60, contains the following parameters:

- Standard Header. Byte 1 indicates the request type. This is X'F0' for an install request.
- Pointer to NETNAME_FIELD, a variable-length field containing the network name of the terminal requesting logon, preceded by a 2-byte length field.
- Pointer to MODELNAME_LIST, an array of names of eligible autoinstall models. The array is preceded by a 2-byte field describing the number of 4-byte name elements in the array. If there are no elements in the array, the number field is set to zero.
- Pointer to SELECTED_PARMS, the area of storage that you use to return information to CICS/400. This area contains:

MODELNAME	4 bytes
------------------	---------

TERMINAL_ID	4 bytes
PRINTER_ID*	4 bytes
ALTPRINTER_ID*	4 bytes
RETURN_CODE	1 byte
PRINTER_NETNAME*	10 bytes
ALTPRINTER_NAME*	10 bytes

* Items marked with an asterisk are retained for compatibility with mainframe-based CICS, but have no function within CICS/400.

- Pointer to **DEVICE_FIELD**, a field containing:

CICS/400 DEVICE_MASK	4 bytes
DEVICE_TYPE	1 byte
TERMINAL_ID	4 bytes

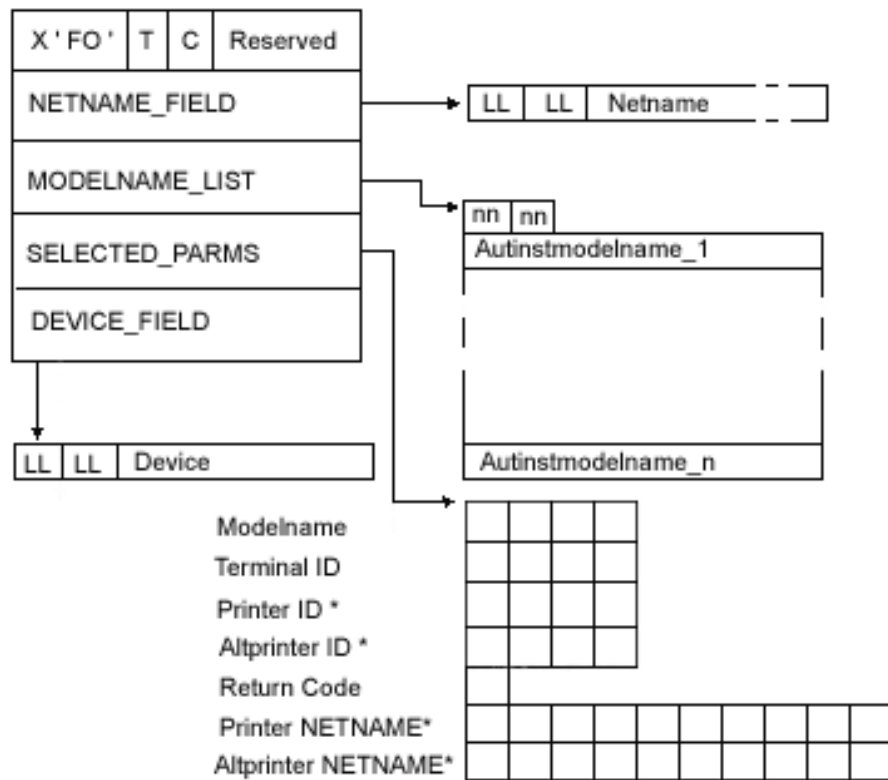
CICS/400 passes a list of eligible autoinstall models in the area addressed by the **MODELNAME_LIST** field of the **COMMAREA**.

The control program must select a model from this list that is suitable for the device logging on, and move the model name to the first 4 bytes of the area addressed by the **SELECTED_PARMS** field of the **COMMAREA**.

The supplied autoinstall control program selects the first model from this list, and CICS/400 uses this model to build the TCTTE for the device.

Before returning to CICS/400, the control program must supply a CICS/400 terminal name for the device logging on, and must set the return code field to X'00' if the autoinstall request is to be allowed.

Figure 61 on page 315 shows all of these fields in their required order.



Note: Items marked with an asterik are retained for compatibility with CICS/ESA, but have no function within CICS/400

LL= logical length

Figure 61. Installation parameter list

Returning information to CICS/400

At installation, the autoinstall control program is responsible for allowing or denying the connection of a new terminal resource to the CICS/400 system. This decision can be based on a number of installation-dependent factors, such as security, or the total number of connected terminals. CICS/400 takes no part in any such checking. You decide whether any such checking takes place, and how it is done.

If the install request is to proceed, the control program **must** do the following:

- Return an autoinstall model name in the first 4 bytes of the area addressed by the SELECTED_PARMS field of the parameter list.

- Supply a CICS/400 terminal name (TERMID) in the next four bytes of the return area.

The CICS/400-supplied AEGTCACP program uses the terminal identifier generated by masking as the terminal name. If you intend to use an autoinstall control program and this does not match your installations's naming conventions, you must code your own autoinstall control program. See "Setting the terminal name".

- Set the return code to X'00'.

On entry to the autoinstall control program, the return code always has a nonzero value. If you do not change this, the autoinstall request is rejected.

Having completed processing, the control program must return to CICS/400 by issuing an EXEC CICS RETURN command.

Selecting the autoinstall model

CICS/400 builds the list of autoinstall models by selecting all the terminal models defined to the CICS system. The complete list of autoinstall models available to CICS/400 at any time comprises all the definitions with DEVMODEL(*MODEL) and DEVMODEL(*BOTH) that have been installed, during CICS system initialization, by the INSCICSGRP CL command, and by CEDA transaction **Install group** command.

CICS/400 gives all the defined models to the autoinstall control program. The CICS/400-supplied autoinstall control program merely picks the first model in the list. However, this model may not provide the attributes required in all cases. Your control program must be able to select the model that provides the characteristics you require for this terminal—for example, automatic task initiation (ATI).

If you need special models for special cases, you can use a simple mapping of, for example, the network name (generic or specific) to the autoinstall model name. Your control program could go through a table of special case network names, choosing the specified model for each. (Note that the list of models presented to the control program is in alphabetical order.)

If CICS/400 does not find any models defined, or the return code in the return area addressed by the SELECTED_PARMS pointer of the parameter list is nonzero, CICS/400 issues an error message and the autoinstall installation fails. The STRCICSUSR command terminates, indicating that autoinstall processing has failed.

Setting the terminal name

The terminal name must be unique, and one through four characters long. The first character must be alphabetic, or one of the special characters \$, @, or #. The remaining characters can be alphanumeric, or the special characters \$, @, or #. The network name may have a maximum length of eight characters. The first character must be alphabetic, or one of the special characters \$, @, or #. The remaining characters can be alphanumeric, or the special characters \$, @, or #. (The terminal name is the identifier CICS/400 uses for the terminal. The network name is the identifier OS/400 uses for the terminal.)

The CICS/400-supplied autoinstall control program creates the terminal name from the terminal identifier generated by masking. This may not satisfy the requirement for uniqueness. One way of overcoming this problem is to use the EXEC CICS

INQUIRE TERMINAL command from the control program, to determine whether the terminal name is already in use. If it is, modify the last character of the terminal identifier and check again.

However, you may be in a situation where you must continue to use unique and predictable terminal names for your terminals. Your control program must be able to assign the right terminal name to each terminal, every time the user logs on.

Two possible approaches to this problem are:

- Devise another algorithm to generate predictable TCT names from network names.

Devising an algorithm avoids the disadvantages of using a table or a file, but it might be difficult to ensure both uniqueness and predictability. If some of the information in the network name is not needed by CICS/400, it can be omitted from the terminal name. An algorithm is probably most appropriate in this situation.

- Use a table or file to map terminal names to network names.

Using a table has two disadvantages, each of which loses you some of the benefits of autoinstall: it takes up storage and it must be maintained.

CICS/400 action on return from the control program

If the return code from the autoinstall program is zero, CICS/400 tries to acquire the device to complete the logon request. If the installation process fails, the control program is called again, as though a delete had occurred. (See the section "Deleting autoinstalled terminal definitions" for details.) This is necessary to allow the program to free any allocations (for example, terminal identifiers) made on the assumption that this install request would succeed.

If the return code is not zero, CICS/400 rejects the installation request in the same way as it rejects an attempt to log on from an unknown terminal when autoinstall is not enabled.

For all autoinstall activity, messages are written to the CSMT log. In addition, autoinstall activity messages are written to the job log of the process attempting the STRCICSUSR request. If an install request fails, a message is issued with a reason code. You can therefore check the output from CSMT and the job log to find out why an autoinstall request failed.

Deleting autoinstalled terminal definitions

The autoinstall control program is also called when:

- A session with a previously automatically-installed terminal has ended.

In this case, CICS/400 issues an internal Interval Control START request for supplied transaction CATD, with the terminal identifier as the data to the request. The CATD transaction is run as a non-terminal related task in an Interval Control batch shell.

- An autoinstall request was accepted by the user program, but the subsequent install process failed for some reason.

In this case, the autoinstall control program is called within the STRCICSUSR process where the autoinstall process has failed.

Invoking the control program at deletion of autoinstalled terminal definitions enables you to reverse the processes carried out at installation. For example, if the

control program at installation incremented a count of the total number of automatically installed resources, the control program at deletion could decrement that count.

The COMMAREA for a delete request

Input to the program is through a COMMAREA.

Data area	Standard header
Byte 1	Request type (X' F1 ' for delete)
Bytes 2-3	Component ID (' TC ')
Bytes 4-16	Reserved
Bytes 17-20	Terminal ID of deleted terminal

Figure 62. COMMAREA at deletion

The COMMAREA, the layout of which is shown in Figure 62, contains the following parameters:

- Standard Header. Byte 1 indicates the request type. This is X'F1' for a delete request.
- The 4-byte terminal identifier of the deleted resource. Figure 63 shows all these fields in their required order.

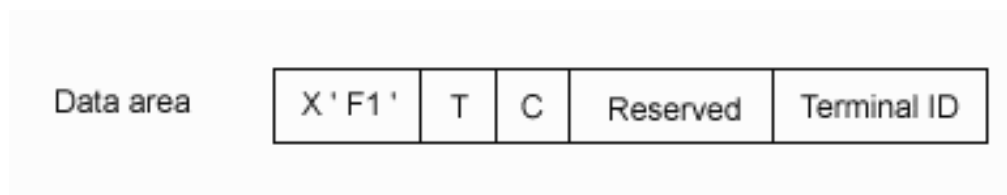


Figure 63. Autoinstall control program's parameter list for delete

Note that the named resource has been deleted by the time the control program is invoked.

Naming your control program

Autoinstall processing is controlled by the program object named in the PPT definition for AEGTCACP. In the PPT definition, the PGMID **must** be AEGTCACP, but the PGMOBJ name may be whatever name you choose. For example:

```
ADDICSTCT LIB(QTEST) GROUP(AUTO) PGMID(AEGTCACP) PGMOBJ(AUTOTEST)
```

Testing and debugging your control program

To help you test the operation of your autoinstall control program, you can run the program as a normal terminal-related application. Define your program and initiate it from a terminal. The parameter list passed to the program is described in "Autoinstalling terminal definitions" on page 313. You can construct a dummy parameter list in your test program, upon which operations can be performed.

You can run your autoinstall control program on a terminal, and use CEDF to test and debug it, provided that the program is not being used for autoinstall

processing. Because there is not a terminal associated with autoinstall processing, it is not possible to use CEDF for testing when the program is invoked as the autoinstall control program. CEDF cannot be invoked, even if the AEGTCACP PPT entry is defined with CICSDEBUG(*DEBUG).

You can also make the test program interactive, sending and receiving data from the terminal. Your autoinstall control program should not perform any terminal related activity.

Customizing the sample program

You can customize the sample program to carry out any processing that suits your installation. Generally, your user program could:

- Count and limit the total number of logged-on terminals.
- Count and limit the number of automatically installed terminals.
- Keep utilization information about specific terminals.
- Map terminal name and network name.
- Do general logging.
- Handle special cases (for example, always allow specific terminals or users to log on).
- Exercise network-wide control over autoinstall. A network-wide, global autoinstall control program can reside on one CICS/400 system. When an autoinstall request is received by a control program on a remote CICS/400 system, this global control program can be invoked and data transferred from one control program to another.

Figure 64 on page 320, in COBOL/400, shows how to redefine the network name, so that the last four characters are used to select a more suitable model than that selected in the sample control program.

```

*
* Redefine the network name so that the last 4 characters (of 10)
* can be used to select the autoinstall model to be used.
*
* The network names to be supplied are known to be of the form:
*
* PFXHVMXNNN
*
* PFXHVM is the prefix
* X      is the system name
* NNN    is the address of the terminal
*
01 NETNAME-BITS.
   02 PREFIX          PIC X(6).
   02 NEXT-CHRS.
   03 NODE-LETTER     PIC X(1).
   03 NODE-ADDRESS    PIC X(3).
   .
   .
PROCEDURE DIVISION.
   .
   .
*
* Select the autoinstall model to be used according to the
* note letter (see above). The models to be used are user
* defined.
*
* (It is assumed that the network name supplied in the COMMAREA
* by CICS/400 has been moved to NETNAME-BITS.)
*
* If the note letter is C, use model AUT1
* If the terminal network name is PFXHVMC289 (a special case), use
* model AUT2.
* Otherwise (node letters A,B,D...), use model AUT3.
*
   IF NODE-LETTER = "C" THEN MOVE "AUT1" TO SELECTED-MODELNAME.
   IF NEXT-CHRS = "C289" THEN MOVE "AUT2" TO SELECTED-MODELNAME.
   IF NODE-LETTER = "A" THEN MOVE "AUT3" TO SELECTED-MODELNAME.
   IF NODE-LETTER = "B" THEN MOVE "AUT3" TO SELECTED-MODELNAME.
   IF NODE-LETTER = "D" THEN MOVE "AUT3" TO SELECTED-MODELNAME.
   .
   .

```

Figure 64. Example of how to customize the AEGTCACP sample program

End of General-use programming interface

Part 4. CICS-supplied transactions

Chapter 12. Introduction to CICS/400-supplied transactions	323
Syntax notation.	323
Online help	323
Levels of access to supplied transactions	323
Role of the system administrator	324

Chapter 13. CEMT—CICS master terminal transaction.	325
Using the CEMT transaction	325
Entering a CEMT command	326
Minimum abbreviation of keywords.	326
Selecting resources	326
Family of resources	327
List of resource identifiers	327
Displaying options and syntax (the ? key).	328
Scrolling data (+ sign)	328
Blank fields in a display.	328
Using the CEMT screens	329
Detecting unprotected fields (the tab key)	330
Program function (PF) keys.	330
Overtyping a CEMT INQUIRE display	331
CEMT DISCARD command	332
CEMT DISCARD	332
CEMT INQUIRE and SET commands	333
CEMT INQUIRE AUTINSTMODEL	333
CEMT INQUIRE SET AUXTRACE	334
CEMT INQUIRE SET CONNECTION	335
CEMT INQUIRE SET FILE.	336
CEMT INQUIRE SET INTTRACE	338
CEMT INQUIRE SET JOURNALNUM.	339
CEMT INQUIRE SET NETNAME	340
CEMT INQUIRE SET PROGRAM	340
CEMT INQUIRE SYSTEM	342
CEMT INQUIRE SET TASK	343
CEMT INQUIRE SET TDQUEUE	344
CEMT INQUIRE SET TERMINAL	346
CEMT INQUIRE SET TRANSACTION.	348
CEMT PERFORM commands	348
CEMT PERFORM SHUTDOWN	349
CEMT PERFORM SNAP	349

Chapter 14. CRTE—CICS routing transaction	351
--	-----

Chapter 12. Introduction to CICS/400-supplied transactions

This chapter introduces:

- “Syntax notation”
- “Online help”
- “Levels of access to supplied transactions”
- “Role of the system administrator” on page 324

CICS/400 supplies a number of transactions that assist you in the running of your system and in application development. See “Defining supplied transactions” on page 65 for a list of all supplied transactions. The transactions that have a terminal operator interface (that is, they must be entered within an active control region environment) are described in the following sections:

- “Using CEDA resource definition online” on page 58
- Chapter 13, “CEMT—CICS master terminal transaction” on page 325
- “CESF—CICS sign-off transaction” on page 129
- Chapter 14, “CRTE—CICS routing transaction” on page 351

The three transactions CEBR, CECI, and CEDF that are used by application programmers to debug their programs are mentioned here because the system administrator needs to allow programmers access to these commands, but the information is intended as an introduction only. These transactions are described in the *CICS for iSeries Application Programming Guide*.

Syntax notation

The syntax display for each supplied transaction uses a standard notation. For an explanation of the command syntax, refer to “Interpreting the syntax diagrams” on page 133.

Depending on the CICS terminal definition, you may be able to enter the transaction identifiers in either uppercase or lowercase characters, although in the examples they are shown in uppercase.

Online help

Online help is available for the CICS/400-supplied transactions with an operator interface. However, the help facility is not available if the transaction is being run remotely using CRTE. To use the help facility, you need to interact with the OS/400 command interface, which you cannot do through CRTE.

Levels of access to supplied transactions

Levels of access to supplied transactions are limited by security levels associated with individual user profiles. Security restrictions are discussed in Chapter 6, “Security requirements for CICS/400” on page 75. Most terminal operators will be restricted to those applications that they use in their working day. This may or may not include CICS-supplied transactions. For example, application programmers need to use the debugging tools CECI, CEBR, and CEDF. Each system, however, should have an administrator who has access to all CICS terminals and CICS-supplied transactions. The CICS/400 system administrator,

known as the master terminal operator on other CICS platforms that support this facility, uses the master terminal transaction CEMT to change control region parameters dynamically.

Role of the system administrator

The system administrator controls CICS control region components using the master terminal transaction CEMT.

Although the CEMT transaction can be started at any valid terminal, its use is intended to be limited. The control permitted through CEMT allows the system administrator to improve performance by changing CICS control region parameters in the day-to-day operation of the CICS control region. In addition, the system administrator has the prime responsibility for administering the facilities of the CICS control region.

By using the routing transaction (CRTE), you can also assume the role of a master terminal operator for other multiple-connected CICS systems. See Chapter 14, “CRTE—CICS routing transaction” on page 351 for details.

As the system administrator, you can access all CICS terminals and supplied transactions. You should understand all the procedures associated exclusively with the master terminal. You should be aware of which terminals and CICS operators can access CICS at any given time, and of the identifiers by which they are known to CICS. For example, when inquiring about CICS terminals, a list of CICS terminals can be supplied (see “CEMT INQUIRE | SET TERMINAL” on page 346).

For LUTYPE6.2 (APPC) connections, the CICS identifier of each parallel session must be known, and you must specify this identifier when operating on the session.

When the CICS control region has satisfactorily completed its response to a CICS command, the time and date are printed or displayed at the terminal, as follows:

```
TIME=hh.mm.ss DATE=yy.ddd
```

Unless otherwise stated, the information about the CICS master terminal and its supplied transactions applies only to a single CICS control region, whether or not it is connected to another CICS system through intersystem communication.

Chapter 13. CEMT—CICS master terminal transaction

This chapter covers the following:

- “Using the CEMT transaction”
- “Entering a CEMT command” on page 326
- “Using the CEMT screens” on page 329
- “Overtyping a CEMT INQUIRE display” on page 331
- “CEMT DISCARD command” on page 332
- “CEMT INQUIRE and SET commands” on page 333
- “CEMT PERFORM commands” on page 348

The CICS CEMT transaction is used to invoke all the master terminal functions, that allow the system administrator to control dynamically the CICS control region. There are four CEMT commands:

CEMT INQUIRE command

Query the status of the specified CICS resources used by the control region.

CEMT SET command

Change the values of parameters used by the control region and alter the status of the CICS control region resources. This command may be used to terminate CICS tasks. No changes are made if there are syntax errors in the CEMT SET command.

CEMT DISCARD command

Remove one or more resource definitions from the control region.

CEMT PERFORM command

Take a control region dump (CEMT PERFORM SNAP) and shut down the CICS control region (CEMT PERFORM SHUTDOWN).

Using the CEMT transaction

To start the master terminal transaction, enter the transaction identifier on the command line:

```
CEMT
```

If you are unfamiliar with CEMT, press the ENTER key and you will be able to use the CEMT screens to guide you through the command. Continue from “Using the CEMT screens” on page 329. Otherwise, type in a full CEMT command, for example:

```
CEMT SET TERMINAL ALL INSERVICE
```

and press the ENTER key. The CEMT transaction checks the syntax of requests, and diagnoses errors. If the request is syntactically correct, it is processed immediately. If the request cannot be processed because of syntax errors, the full syntax of the request is displayed.

When the CICS CEMT transaction has been initiated, CEMT does not need to be reentered on subsequent requests because the identifier is implied at the beginning of any further request.

CEMT

If you type CEMT INQUIRE without any operands, INQUIRE SYSTEM is assumed and a display of the CICS control region parameters is shown. Full details are given in "CEMT INQUIRE SYSTEM" on page 342.

Entering a CEMT command

There are a number of conventions for entering CEMT commands that reduce the amount of information you have to type and also allow you to select the resources you wish to work with.

Minimum abbreviation of keywords

The CEMT transaction accepts as few characters of a keyword as are needed to identify it uniquely within the request. This means that the keyword TASK can be entered as TA or TAS, but T cannot be used, because it can be confused with TDQUEUE, TERMINAL, or TRANSACTION. The first keyword containing the abbreviation is used if it is not unique. An exception to this rule is SHUTDOWN, for which at least SHUT must be specified.

In the syntax displays on the screen, and in some examples in this chapter, the minimum permitted abbreviation is given in uppercase characters, and the remainder in lowercase characters. See Figure 67 on page 330 for the list of abbreviations.

Selecting resources

You can affect either a single resource or a group of resources. For example,
CEMT INQUIRE JOURNALNUM(14) CLOSED

inquires about a single resource, journal 14. If CICS journal 14 is open for output, the command results in the NOTFND response, because the CEMT transaction looks for a CICS journal numbered 14 that is closed.

Many of the request formats contain the ALL option. For CEMT INQUIRE, this is a default that is assumed if an alternative is not specified. For example, if inquiring about CICS terminals, information about all CICS terminals is displayed, unless a CICS terminal identifier is specified.

The ALL option is not the default for CEMT SET commands. If the ALL option on a CEMT SET option is specified, any changes requested are made to all CICS resources of the specified type. If there are no CICS resources of the type specified on the CEMT SET command, a response of NOTFND is given.

For example:

```
INQUIRE TERMINAL ALL INSERVICE
```

displays the status of all CICS terminals that are in service.

If the subdefined CICS resource group has no members, the CEMT transaction returns a NOTFND response. For example, if all the CICS files in the CICS control region are closed, the command:

```
INQUIRE FILE ALL OPEN
```

gives the NOTFND response.

A CICS resource group cannot be subdefined on the CEMT SET command in the same way. For example:

```
SET TERMINAL ALL INSERVICE ACQUIRED
```

puts all CICS terminals both in service **and** into an acquired status.

However, such groups of CICS resources can be managed by issuing a CEMT INQUIRE request, and overtyping the display produced. See “Overtyping a CEMT INQUIRE display” on page 331.

Family of resources

The symbols * and + can be used as part of an identifier to specify a family of CICS resources. The symbol * represents any number of characters (including none), and + represents a single character, for example:

A* All identifiers beginning with A

DATA

All identifiers containing the characters DATA

TERM00+

All 7-character identifiers starting with TERM00

Use of * and + is restricted to selecting existing CICS resources and cannot be used to discard CICS resources.

```
SET TERMINAL(TRM*) INSERVICE
```

sets all CICS terminals, whose identifiers start with TRM, in service.

List of resource identifiers

You can specify a list of CICS resource identifiers, rather than a single one, separating the items in the list by commas or blanks. Any errors of syntax are reported.

```
SET TERMINAL(T01,T02,T03,T04) INSERVICE
```

The symbols * or + cannot be used when the identifier is in a list of identifiers.

Most of the requests either inquire about (INQUIRE) or change (SET) the status of one or more CICS resources (such as a terminal), a particular subgroup of CICS resources (such as terminals whose ids all start with the same characters), or all CICS resources of a particular type (such as all terminals).

On every CEMT SET command, an argument, a generic argument, or ALL must be specified, for example:

```
SET TERMINAL(TRM1) INSERVICE
```

sets CICS terminal TRM1 in service. The command:

```
SET TERMINAL ALL INSERVICE
```

sets all CICS terminals in service. Note that the command:

```
SET TERMINAL INSERVICE
```

produces an error.

Displaying options and syntax (the ? key)

If you type the ? key in the space on the extreme left of the command line, the syntax for that command appears on the screen. For example, ?INQUIRE TERMINAL gives the list of options that you can use the inquiry command on, with the default highlighted.

Following the inquiry command, you can see a list of the resources installed in your control region. For example, when you place the cursor at the beginning of one of the lines of status information, and type a question mark (?), you see a display consisting of that line of data and the syntax of the appropriate CEMT SET command. This is shown in Figure 65.

```

SET TERMINAL
SYNTAX OF SET COMMAND
Ter(00EL) Tra(CEMT) Ins Ati Tti Net(IGKS00EL) Acq

CEMT Set TErminAl()
< ALl >
< REMote system (sysid) >
< Inservice | Outservice >
< ATi | NOAti >
< TTi | NOTTi >
< Purge >
< ACquired | REleased >

RESPONSE: NORMAL                                APPLID - CARINA
PF:  1 HELP  3 END                               TIME: 11.55.55  DATE: 94.146
                                                9 MSG

```

Figure 65. Sample of screen showing the syntax of a CEMT SET command

Scrolling data (+ sign)

A plus (+) sign on the first or last line of a display tells you that there is more data above or below your current display. Scrolling backward reveals data above, and scrolling forward reveals data below.

Blank fields in a display

Some displays contain blank fields that allow you to specify options, such as PURGE on the CEMT SET TERMINAL command. This is not part of the status of a CICS resource.

A status beginning with "NO", for example NOREAD, is not displayed and appears as a blank field. To alter the status, for example from NOREAD to READ, simply type in the change in the blank field. To position the cursor at one of these fields, use either the tab key or the cursor control keys.

Using the CEMT screens

If you type CEMT with no operands and press the ENTER key, you will see a list of the keywords (Discard, Inquire, Perform, and Set), as shown in Figure 66.

```
STATUS: Enter one of the following

Discard
Inquire
Perform
Set

RESPONSE: NORMAL                APPLID - CARINA
PF:  1 HELP  3 END              TIME: 11.56.17  DATE: 94.146
                                   9 MSG
```

Figure 66. CICS CEMT transaction: initial screen

This display prompts you to enter one of the keywords shown; you need enter only the first letter. If INQUIRE is entered, a list of options is shown. See Figure 67 on page 330.

```

INQUIRE
STATUS: Enter one of the following or hit enter for default

AUTinstmodel
AUXtrace
Connection
File
Intrace
Journalnum
Netname
Program
System
TAsk
TDqueue
TERminal
TRansaction

RESPONSE: NORMAL                                APPLID - CARINA
PF:  1 HELP  3 END                               TIME: 11.57.36  DATE: 94.146
                                           9 MSG

```

Figure 67. Sample of the screen following the CEMT INQUIRE command

You can inquire about any of the displayed options by typing the keyword after INQUIRE on the command line. For example:

```
INQUIRE PROGRAM
```

gives you the status of all CICS programs, and for each CICS program gives its attributes. An example is shown in Figure 68 on page 331. Full details are given in “CEMT INQUIRE | SET PROGRAM” on page 340.

Detecting unprotected fields (the tab key)

The fields you can change are different in each display. You can detect them, however, by pressing the tab key repeatedly. This causes the cursor to jump from one unprotected field to the next.

Program function (PF) keys

Each CEMT screen shows at the bottom which function keys are in operation for that screen. The functions of the PF keys are:

- PF1** displays a help panel that describes the usage of the CICS CEMT transaction, by using the OS/400 online facility.
- PF3** ends this master terminal session by terminating the CICS CEMT transaction. If, however, this key is used when the display is being modified, perhaps by typing in a new command, or by overtyping an old command, this key does not end the session; it is ignored.
- PF7** scrolls backward half a screen.
- PF8** scrolls forward half a screen.
- PF9** displays up to 20 messages on a separate panel. If more than one message has been generated in response to the request, the number of messages generated appears near the bottom of the screen. Press ENTER to go back to the original panel.

PF10 scrolls backward a full screen.

PF11 scrolls forward a full screen.

Overtyping a CEMT INQUIRE display

When you make an inquiry, you usually get a display that consists of status information for each CICS resource in the specified group. See Figure 68.

The status information is displayed as a list of abbreviated keywords. The

```

INQ PROG
STATUS: Results type over to change

- Pro (ACCTSET ) Use (000003) Res (000) Cob Map   Ena
  Library (QCICSSAMP ) Object (ACCTSET ) Len (0008192)
? Pro (ACCT00 ) Use (000006) Res (001) Cob Pro Ced Ena Fu1 New
  Library (QCICSSAMP ) Object (ACCT00 ) Len (0020480)
- Pro (ACCT01 ) Use (000003) Res (000) Cob Pro Ced Ena Fu1
  Library (QCICSSAMP ) Object (ACCT01 ) Len (0089088)
- Pro (ACCT02 ) Use (000000) Res (000) Cob Pro Ced Ena Fu1
  Library (QCICSSAMP ) Object (ACCT02 ) Len (0091136)
- Pro (ACCT03 ) Use (000000) Res (000) Cob Pro Ced Ena Fu1
  Library (QCICSSAMP ) Object (ACCT03 ) Len (0053760)
- Pro (ACCT04 ) Use (000000) Res (000) Cob Pro Ced Ena Fu1
  Library (QCICSSAMP ) Object (ACCT04 ) Len (0092672)
- Pro (AEGGMSG) Use (000001) Res (000) Cob Pro   Ena Fu1
  Library (QCICSSAMP ) Object (AEGGMSG ) Len (0041984)
+ Pro (DFH$DGA ) Use (000000) Res (000) Cob Map   Ena
  Library (QCICSSAMP ) Object (DFH$DGA ) Len (0001536)

                                         APPLID - CARINA
RESPONSE: NORMAL                       TIME: 11.57.59  DATE: 94.146
PF:  1 HELP  3 END  7 SBH  8 SFH  9 MSG 10 SB 11 SF

```

Figure 68. Example of a CEMT status display

highlighted entries on the third line of the display show the positions of the input fields and possible contents. You can move the cursor to these fields and change their contents by overtyping.

Table 27. Explanation of CEMT status display input fields

Example contents	Meaning	Explanation
?	SET syntax or discard program	Enter ? to see the SET syntax, or d to discard the program entry. You can use this as an alternative to the CEMT DISCARD command, either if you do not know the resource name or if you wish to discard more than one resource. See “CEMT DISCARD command” on page 332 for details of the CEMT DISCARD command.
Ced	CEDF/NOCEDF	Enter C if CEDF is allowed, N if CEDF is not allowed.
Ena	Enabled/disabled	Enter E if the entry is to be enabled, or D if the entry is not enabled.

CEMT

Table 27. Explanation of CEMT status display input fields (continued)

Example contents	Meaning	Explanation
Ful	Full API/DPL subset	Enter F to request the full API, or D to request the DPL API subset. If the entry is a remote program or a map, this field is blank.
New	Newcopy/phasein	Enter N to use the new copy of the program object when it is next referenced, or P to use the new program object immediately.

When you press ENTER again, CICS reads the contents of all fields that have been changed, and processes any valid operations implied by the changes. You need to change only the first character of the field, because CEMT checks only the number of characters required to identify the request. For example, to change Ena to Dis, you need type only D. CEMT treats Dna as if Dis had been entered. For this reason, you must make sure that you type the changes in the correct input fields. For example, if you want a new copy of a program, but type N in the CEDF/NO CEDF field, CEMT will read the N as meaning NOCEDF.

If you make an incorrect change, you get an error message, and the field is not changed.

Whenever you overtype a display, not only is that particular action taken but all the status information is refreshed. You can avoid the overhead of a large number of locates either by using the CEMT SET command, or by limiting your inquiry to a specific number of CICS resources.

CEMT DISCARD command

You use the CEMT DISCARD command to remove certain resources from the control region. These resources are not deleted, and will be available again when the control region is restarted.

CEMT DISCARD

Format

```
▶▶ CEMT Discard Autinstmodel—(—4-char data-value—)
  File—(—8-char data-value—)
  Program—(—8-char data-value—)
  Transaction—(—4-char data-value—)◀◀
```

Purpose

The CEMT DISCARD command allows you to remove an installed CICS resource definition from the CICS control region for the duration of the session.

DISCARD does not affect the CICS resource definition on the OS/400, and therefore the CICS resource can be reinstated using the CEDA INSTALL command or the OS/400 INSCICSTBL CL command. CEMT DISCARD commands have the same security attached to them as CEMT SET commands. You cannot discard CICS resources that are currently in use, or that are CICS owned resources (beginning with "AEG" or "C").

Note: You can use the CEMT INQ command to display a list of resources, from which you can then discard selected resources. A CEMT INQ display can be used to discard more than one entry at a time, but you cannot use it to disable and discard at the same time. See “Overtyping a CEMT INQUIRE display” on page 331 for further details.

Parameters

AUTINSTMODEL

The name (1–4 characters) of the CICS autoinstalled terminal model as specified by the CICSDEV parameter of the TCT. For details, see “Defining and maintaining group definitions” on page 45.

FILE

The name of the CICS file as specified by the FILEID parameter of the FCT.

PROGRAM

The name of the CICS program or map set as specified by the PGMID parameter of the PPT.

TRANSACTION

The name of the CICS transaction as specified by the TRANSID parameter of the PCT.

Notes:

1. You cannot discard a CICS program if there is an installed CICS transaction definition that refers to it. The CICS transaction must be discarded first, then the CICS program.
2. You cannot discard a CICS remote file.
3. When you discard a CICS resource, you cannot use * or + in the CICS resource name.

CEMT INQUIRE and SET commands

Descriptions of the combined CEMT INQUIRE and SET commands follow in alphabetic order of the name of the CICS resource. The options of each command are also described in alphabetic order.

Further information on the errors and restrictions in the use of the CEMT SET commands may be found in the *CICS for iSeries Application Programming Guide*.

CEMT INQUIRE AUTINSTMODEL

▶▶—CEMT Inq AUTinstmodel—┐
└(—autoinstall-model-name—)┘▶▶

Function

The INQUIRE AUTINSTMODEL command retrieves information about CICS autoinstall terminal model definitions in the control region. If you supply an autoinstall model name, details of that model are displayed. If you omit an autoinstall model name, details of all models in the control region are displayed. For more information about autoinstall, see Chapter 11, “Autoinstall for terminal definitions” on page 305.

There is no SET AUTINSTMODEL function. The INQUIRE AUTINSTMODEL screen also allows the use of the DISCARD AUTINSTMODEL function.

CEMT INQUIRE AUTINSTMODEL

The status line shows the following information:

- The name of the CICS autoinstall terminal model Auti (autoinstall-model-name).
- Whether the CICS autoinstall terminal model is available for use (Ena) or not (Dis).

Required parameters

AUTINSTMODEL

The name (1–4 characters) of the autoinstall terminal model as specified by the CICSDEV parameter of the TCT.

CEMT INQUIRE/SET AUXTRACE

▶▶—CEMT Inq AUXtrace—▶▶

▶▶—CEMT Set AUXtrace—▶▶

—STArt— —STOp—	—Noswitch— —All—	—SWItch—
-------------------	---------------------	----------

Function

The INQUIRE AUXTRACE command retrieves information that controls the recording of the CICS auxiliary trace entries in the CICS control region.

The SET AUXTRACE command controls the recording of CICS auxiliary trace entries in the CICS control region.

The status line shows the following information:

- Which of the CICS auxiliary trace user spaces is active: CUR (A) or CUR (B).
- Whether the CICS auxiliary trace is active (Sta) or not (Sto).
- What the CICS auxiliary trace does when the CICS auxiliary trace user space is full: switch to the next CICS auxiliary trace user space automatically (All) or stop the CICS auxiliary trace (Nos).

Required parameters

ALL

Automatic switching between the two CICS auxiliary trace user spaces is to occur as necessary until the end of the CICS control region, without the need for operator intervention.

NOSWITCH

When the CICS auxiliary trace user space is filled, the CICS auxiliary trace is stopped.

START

CICS auxiliary tracing is to be started.

STOP

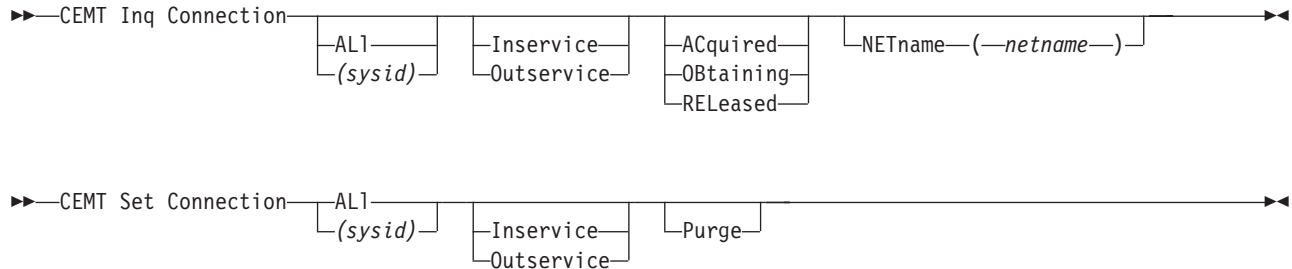
CICS auxiliary tracing is to stop. A subsequent START request causes new trace entries to be written at the start of the CICS auxiliary trace user space, thereby overwriting the trace entries that were written before the STOP request.

SWITCH

CICS auxiliary trace user space is switched to the alternate CICS auxiliary trace user space.

For more information about traces, see the *CICS for iSeries Application Programming Guide* and *CICS for iSeries Problem Determination*.

CEMT INQUIRESET CONNECTION



Function

The INQUIRE CONNECTION command retrieves information about CICS connection definitions (sometimes known as “system entries”) in the CICS control region.

The SET CONNECTION command changes the connection definition in the CICS control region.

The status line shows the following information:

- The name of the CICS connection, CONN(sysid).
- Whether the CICS connection is available for use (Ins) or not (Out).
- Whether the CICS connection has a bound session (Acq), is in the process of obtaining a bound session (Obt), or the CICS connection is released (Rel).
- The CICS net name associated with the CICS connection, NET(netname).

Required parameters

ACQUIRED

The connection is ACQUIRED when it meets two criteria:

- The partner LU has been contacted
- Initial CNOS exchange has been done

APPC sessions are acquired; sessions that would be established at CICS startup are acquired. This operand is ignored if the system is out of service. For further information on APPC session acquisition, see *CICS for iSeries Intercommunication*.

ALL

All CICS connection entries are checked.

CONNECTION(sysid)

This is the name (1–4 characters) of the other CICS system as specified by the CTRLGN parameter of the TCS.

INSERVICE

CICS connection is made available for use. This option is included for compatibility with other members of the CICS family and indicates the CICS status only. Use the OS/400 command set to interrogate the true connection status.

CEMT INQUIRE/SET CONNECTION

NETNAME(*netname*)

This is the name (1–8 characters) by which the remote system is known in the network as specified by the NETWORK parameter of the TCS.

OBTAINING

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

OUTSERVICE

The CICS connection is not available for use.

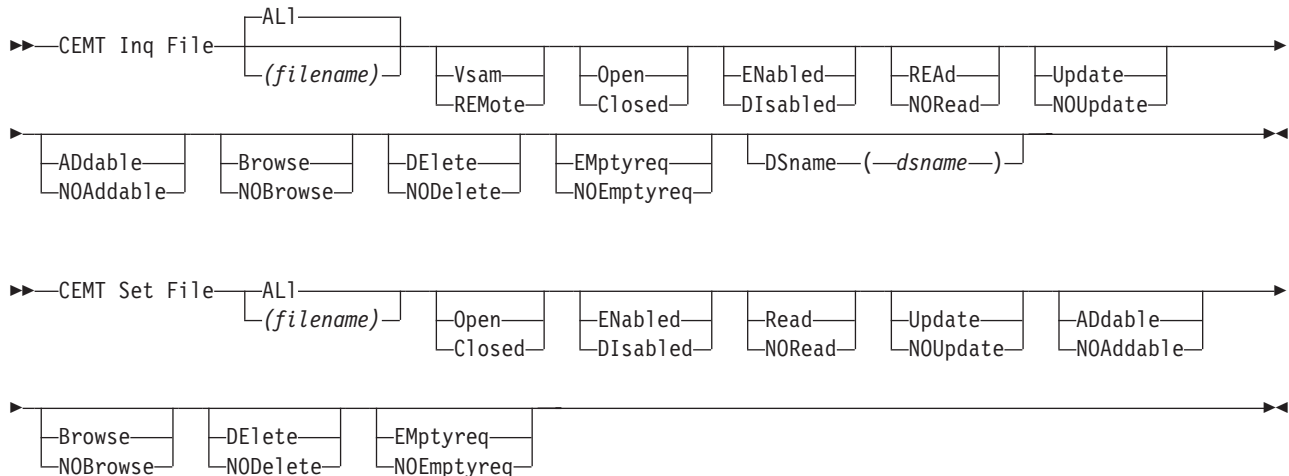
PURGE

CICS tasks associated with the CICS connection are abnormally terminated, but CICS task termination occurs only if system and data integrity can be maintained.

RELEASED

The CICS connection is released.

CEMT INQUIRE/SET FILE



Function

The INQUIRE FILE command retrieves information about CICS file definitions in the CICS control region. You are allowed to use the DISCARD FILE function from the INQUIRE FILE screen.

Note: The value that is returned varies according to the sequence in which the command is issued compared to the file status. For example, if the CICS file is closed when the command is issued, much of the information received shows the state the CICS file will be in when it is next opened. If the CICS file has never been opened, default or null values are received for some of the options. Those values could change when the CICS file is opened.

The SET FILE command changes some of the information of a CICS file definition in the CICS control region.

The status line shows the following information about the CICS file, displayed positionally:

- The name of the file (Fil (*filename*)).
- Whether or not it is remote (Vsa or Rem).

- Whether it is available for use (Ena or Dis).
- Whether the file can be read (Rea or Nor).
- Whether it can be updated (Upd or Nou).
- Whether records can be added (Add or Noa).
- Whether it can be browsed (Bro or Nob).
- Whether it can have records deleted (Del or Nod).
- Whether the file can be cleared when it is opened (Emp or a blank).
- The name of the file object that is associated with the CICS file:
Library(library-name), File(file-name), and Member(member-name).

Required parameters

ADDABLE

This allows records to be added to the CICS file.

ALL

All CICS files are checked.

BROWSE

This allows records to be browsed in the CICS file.

CLOSED

The CICS file is to be closed.

The close is effected at the time of the command only if there are no CICS tasks currently accessing the CICS file. If there are current CICS users, the display indicates CLOSE pending. This means that the close is effected when the last CICS user finishes using the CICS file.

DSNAME(dsname)

This is the name (1–10 characters) of the OS/400 file object with which this file is associated, as specified by the FILE and MBR parameters of the FCT.

DELETE

This allows records to be deleted from the CICS file.

DISABLED

The CICS file is not available for use by CICS application programs.

The disable is effected at the time of the command only if there are no CICS tasks currently accessing the CICS file. If there are current CICS users, the display indicates DISABLE pending. This means that the disable is effected when the last CICS user finishes using the CICS file.

EMPTYREQ

The CICS file is cleared when open. It indicates that, when a CICS file, to which the OS/400 file object is allocated, is next opened, its data is erased.

ENABLED

The CICS file is now available for use by CICS application programs and is opened on the first request.

filename

This is the name (1–8 characters) of the CICS file as specified by the FILEID parameter of the FCT.

NOADDABLE

Records are not allowed to be added to the CICS file.

NOBROWSE

Records are not allowed to be browsed in the CICS file.

CEMT INQUIRE|SET FILE

NODELETE

Records are not allowed to be deleted from the CICS file.

NOEMPTYREQ

The CICS file is not to be cleared when open. Applies only to OS/400 file objects that have been defined with the REUSE parameter. When the CICS file is next opened, its data is not erased.

NOREAD

Records are not allowed to be read in the CICS file.

NOUPDATE

Records are not allowed to be updated in the CICS file.

OPEN

The CICS file is to be opened.

READ

This allows records to be read in the CICS file.

REMOTE

The CICS file resides on another CICS system.

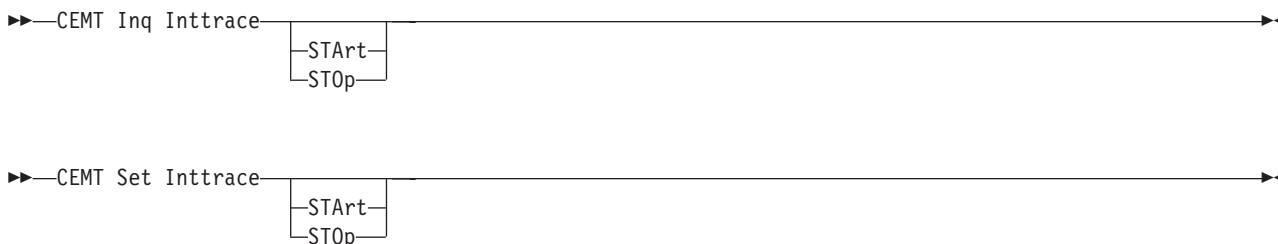
UPDATE

This allow records to be updated in the CICS file.

VSAM

The CICS file emulates the virtual storage access method (VSAM).

CEMT INQUIRE|SET INTTRACE



Function

The INQUIRE INTTRACE command retrieves information that controls the recording of CICS internal trace entries in the CICS control region.

The SET INTTRACE command changes the recording of CICS internal trace entries in the CICS control region.

The information that is displayed on the status line shows whether the CICS internal tracing is active (Sta) or not (Sto).

Required parameters

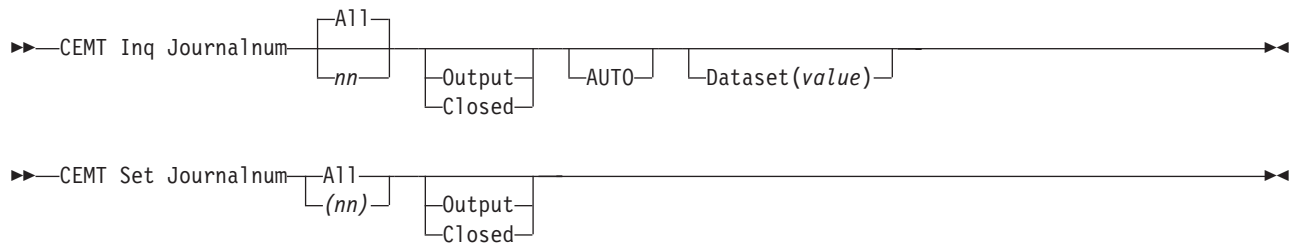
START

CICS internal tracing is to start.

STOP

CICS internal tracing is to stop.

CEMT INQUIRE|SET JOURNALNUM



Function

Note: A list of identifiers cannot be specified, nor can the symbols * and + be used to specify a family of CICS journals.

The INQUIRE JOURNALNUM command retrieves information about CICS journal definitions in the CICS control region.

The SET JOURNALNUM command changes the journal definition in the CICS control region.

The status line shows the following information:

- The number of the CICS journal, Jrn(nn).
- Whether the CICS journal is open (Ope) or closed (Clo).
- Whether or not the CICS journal is to switch automatically (Auto) when it is full.
- The name of the OS/400 file object that is associated with the CICS journal: Library(library-name), File(file-name), and Member(member-name), as specified in the JCT.

Required parameters

ALL

All CICS journals are checked.

AUTO

The CICS journal will switch automatically when full to the next generation.

CLOSED

The CICS journal is to be closed.

DATASET(value)

This is the name (1–32 characters) of the OS/400 physical file with which this CICS journal is associated, as specified by the JRNLIB, JRNFILE, and JRNMBR parameters of the JCT.

JOURNALNUM(nn)

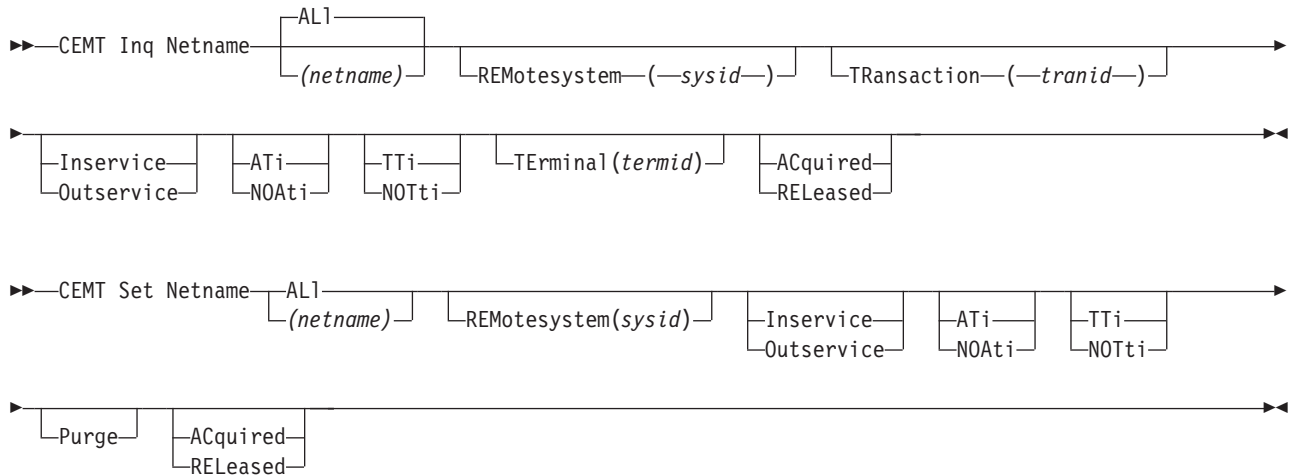
This is the number (1–2 digits) of the CICS journal as specified by the JFILE parameter of the JCT.

OUTPUT

The CICS journal is opened for output.

CEMT INQUIRE|SET NETNAME

CEMT INQUIRE|SET NETNAME



Function

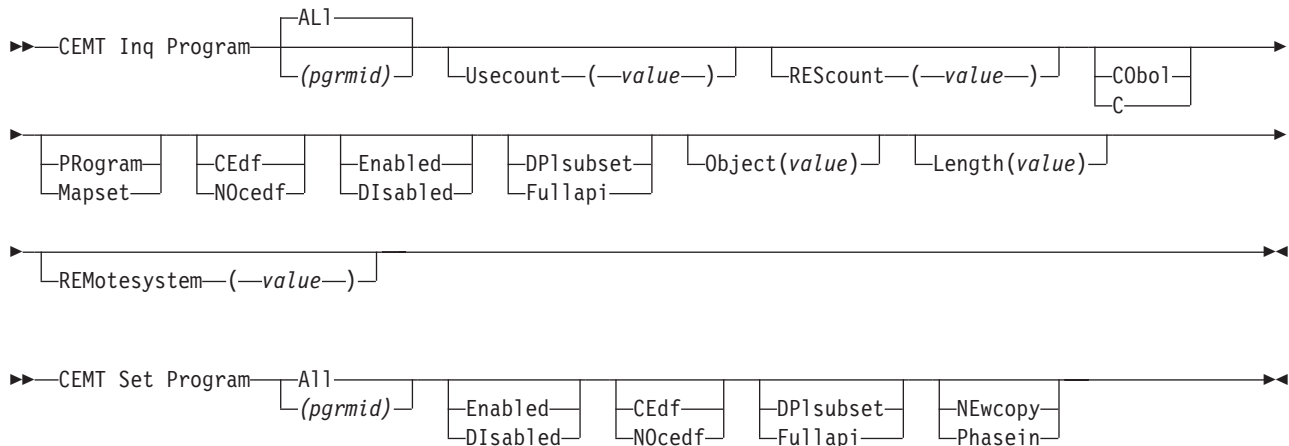
If network names are familiar, you may prefer this command. It is an alternative to `TERMINAL`, as described under “`CEMT INQUIRE|SET TERMINAL`” on page 346.

Required parameters

The status line shows the following information:

- The netname (Net (netname)).
- The CICS transaction ID that is associated with the CICS terminal (Tra (tranid)).
- Whether the CICS terminal is in service (Ins) or not (Out).
- Whether the CICS terminal has ATI ability (Ati) or not (a blank).
- Whether the CICS terminal has TTI ability (Tti) or not (a blank).
- Whether the CICS terminal is acquired (Acq) or not (Rel).

CEMT INQUIRE|SET PROGRAM



Processing

The `INQUIRE PROGRAM` command retrieves information about CICS program definitions in the CICS control region. The `INQUIRE PROGRAM` screen also allows the use of `DISCARD PROGRAM` function.

CEMT INQUIRE/SET PROGRAM

The SET PROGRAM command changes the program definition in the CICS control region.

The status line shows the following information:

- The name of the CICS program or map, Pro(pgrmid).
- How many times the CICS program has been used, since it was loaded by CICS, Use (value).
- How many CICS users are currently using the CICS program, Res (value).
- Whether the calling convention of the program is COBOL/400 (Cob) or ILE C.
- Whether the CICS program is a map (Map) or a program (Pro).
- Whether CEDF can be used (Ced) by the CICS program (Ced) or not (a blank).
- Whether the CICS program is enabled (Ena) or not (Dis).
- Whether CICS is to use a new copy of the program when all current transactions have finished (New) or immediately (Pha).
- The name of the OS/400 object that is associated with the CICS program: Library(library-name) and Object(object-name).
- The size of the object code associated with the CICS program, Len (value).

Required parameters

ALL

All CICS programs are checked. Cannot be specified if pgrmid is used.

C CICS/400 uses the CICS C calling convention to invoke the OS/400 program object.

CEDF

The CICS program allows EDF to intercept all CEDF activity. See the *CICS for iSeries Application Programming Guide* for more information about EDF. This includes initiation and termination screens while this CICS program is being processed, unless the CICS program was translated using the NODEBUG translator option.

COBOL

CICS/400 uses the CICS COBOL calling convention to invoke the OS/400 program object.

OBJECT(value)

This is the name (1–10 characters) of the OS/400 program with which this CICS program is associated, as specified by the PGMOBJ parameter of the PPT.

DISABLED

The CICS program is not available for use. Programs beginning with “AEG” cannot be disabled because these characters are reserved for use by CICS.

DPLSUBSET

The CICS program can only use the distributed program link subset of CICS commands.

Note: This option cannot be specified for a program beginning with “AEG”.

ENABLED

The CICS program is to be available for use.

FULLAPI

The CICS program can use the full set of application program interface (API) commands that are available with CICS/400.

CEMT INQUIRE|SET PROGRAM

LENGTH(value)

This is the size of the CICS program in bytes. The value returned is zeros, if the CICS program has not been loaded by the CICS control region.

MAPSET

The CICS program is defined as a map set.

NEWCOPY

CICS is to use a new copy of the program. This option may be used only when no transactions are using the program in questions. A response of RESCOUNT > 0 is displayed if the program is in use.

NOCEDF

The CICS program is not to allow EDF to intercept. All CEDF activity, including initiation and termination screens, stops while this CICS program is being processed.

PHASEIN

CICS is to use a new copy of the program now, for all new transaction requests. CICS continues to use the old copy for all currently-running transactions, until they have finished (RESCOUNT equal to zero). CICS loads the new version, resolving the object as defined by the PGMOBJ parameter of the PPT entry.

PROGRAM

The CICS program is defined as an application program.

PROGRAM(pgrmid)

This is the name (1–8 characters) of the CICS program as specified by the PGMID parameter of the PPT.

REMOTESYSTEM(value)

The name (4 characters) of the associated remote system is returned.

RESCOUNT(value)

This is the number of times the CICS program is currently being used.

USECOUNT(value)

This is the total number of times the CICS program has been used since the CICS program was loaded by the CICS control region.

CEMT INQUIRE SYSTEM

Format

▶▶—CEMT Inq System—◀◀

Purpose

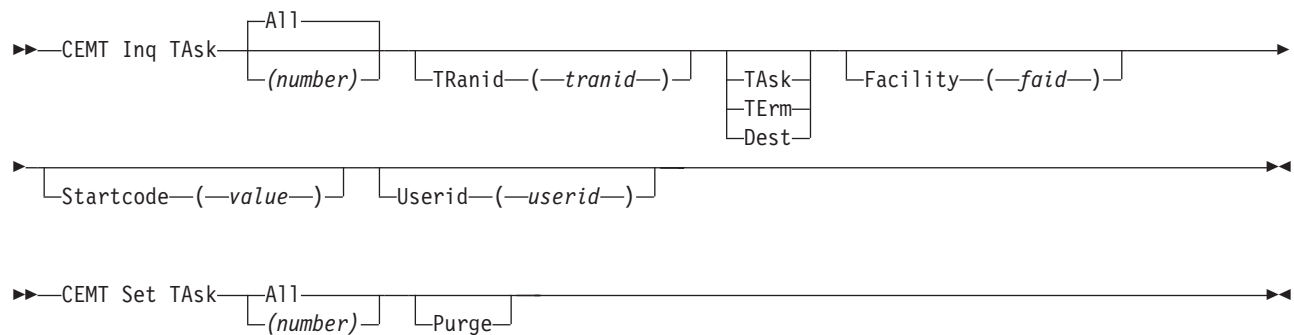
The INQUIRE SYSTEM command retrieves information about the CICS control region.

The SYSTEM keyword does not have to be coded. If you enter just CEMT INQ, SYSTEM is assumed. There is no SET SYSTEM function.

The status line shows the following information:

- The formal release number of the operating system currently running, OPRel (value).
- The type of operating system currently running, OPSys (value).
- The level of CICS present, Release (value).

CEMT INQUIRESET TASK



Function

Note: A list of identifiers cannot be specified, nor can the symbols * and + be used to specify a family of CICS/400 tasks.

The INQUIRE TASK command retrieves information about CICS user tasks in the CICS control region. Information about CICS user tasks can be displayed or changed. The mirror transaction CPMT can also be displayed. Information about CICS/400-generated system tasks or subtasks cannot be displayed or changed. CICS/400 system tasks are those tasks started (and used internally) by CICS/400, and not as a result of a CICS/400 user transaction.

The SET TASK PURGE command purges a CICS/400 user task in the control region.

The status line shows the following information:

- The identifier of the CICS task, Tas (number).
- The CICS transaction associated with the CICS task, Tra (tranid).
- The CICS facility that is associated with the CICS task, Fac (faid).
- Whether the user task has been started from a terminal (Ter) or not (a blank).
- The start code associated with the CICS task.

Required parameters

ALL

All CICS user tasks are checked.

DEST

The CICS/400 user task has been initiated by a destination trigger level.

FACILITY(faid)

This is the identifier of the CICS/400 terminal or queue that initiated the CICS/400 user task. If no FACILITY value is displayed, the CICS/400 user task was started without a facility.

PURGE

This terminates the CICS/400 user task. Termination occurs only when system and data integrity can be maintained.

STARTCODE(value)

This is a code indicating how this CICS/400 user task was started. The possible values are:

CEMT INQUIRESET TASK

D	Distributed program link (DPL)
DS	DPL plus sync-on-return
QD	Transient data trigger level was reached
S	Start command (no data)
SD	Start command (with data)
TO	Operator entered a CICS/400 transaction at the terminal
TP	A CICS/400 transaction was started by presetting the CICS/400 transaction for the terminal
U	User-attached CICS/400 task

TASK

The CICS/400 user task has been initiated from another CICS/400 user task.

TASK(number)

This is the number of the CICS/400 user task.

TERM

The CICS/400 user task has been initiated from a CICS/400 terminal.

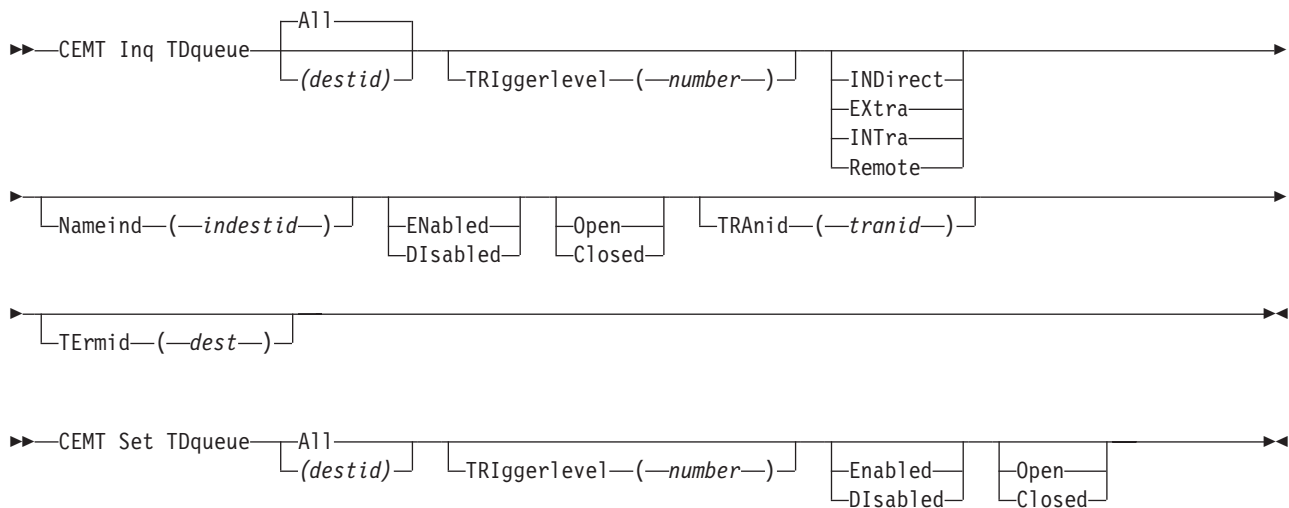
TRANID(tranid)

This is the CICS/400 transaction name associated with the CICS/400 user task.

USERID(userid)

This is the identifier (1–10 characters) of the CICS/400 user currently associated with the CICS/400 task.

CEMT INQUIRESET TDQUEUE



Function

The INQUIRE TDQUEUE command retrieves information about CICS transient data queue definitions in the CICS control region. For information about transient data queues, see the *CICS for iSeries Application Programming Guide*.

The SET TDQUEUE command changes the transient data queue definition in the CICS control region. The queue must not be REMOTE or INDIRECT.

The status line shows the following information:

- The name of the CICS TD queue, TDqueue (destid).
- Whether the CICS TD queue is indirect (Ind), extrapartition (Ext), intrapartition (Int), or remote (Rem).
- Whether the CICS TD queue is enabled (Ena) or not (Dis).
- Whether the CICS TD queue is open (Ope) or not (Clo).
- If the CICS TD queue is indirect, the indirect destid (NAMEIND indestid) is shown.
- If the CICS TD queue is intrapartition, the trigger-level (TrigLvl number) is shown. If the trigger level is not zero, the CICS transaction ID (Tra (tranid)), and the CICS terminal ID (Ter (dest)) are also shown.

Required parameters**ALL**

All CICS TD queues are checked.

CLOSED

The CICS extrapartition queue is to be closed.

DISABLED

The CICS TD queue is not available for use. Queues beginning with "C" cannot be disabled because they are usually reserved for use by CICS.

ENABLED

The CICS TD queue is available for use.

EXTRA

The CICS TD queue type is extrapartition.

INDIRECT

The CICS TD queue type is indirect.

INTRA

The CICS TD queue type is intrapartition.

NAMEIND (Indirect queues only)

This is the name (1–4 characters) of the CICS TD queue that the INDIRECT queue points to, as specified by the PHYDEST parameter of the DCT.

OPEN

The CICS extrapartition queue is to be opened.

REMOTE

The CICS TD queue type is remote.

TDQUEUE(destid)

This is the name (1–4 characters) of the CICS TD queue as specified by the DEST parameter of the DCT.

TERMID(dest)

This is the name (1–4 characters) of the CICS terminal or session to be associated with this CICS TD queue when automatic transaction initiation occurs. Also see the TRANID and TRIGGERLEVEL options.

TRANID(tranid)

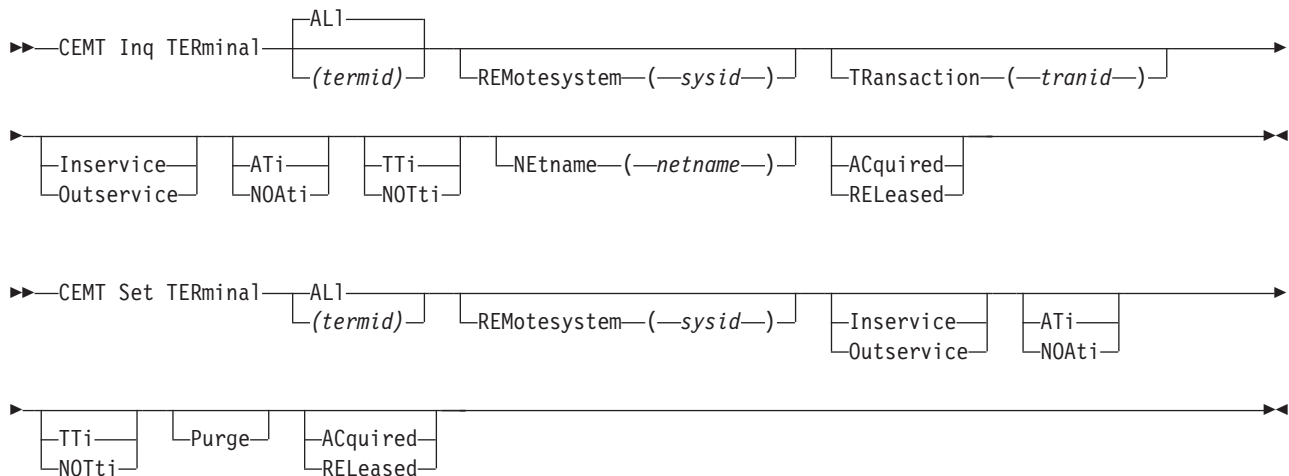
This is the identifier (1–4 characters) of the CICS transaction that is to be initiated automatically when the CICS TD queue trigger level is reached.

TRIGGERLEVEL(number)

This is the number (0–32 767) of requests for output to the CICS TD queue that must be reached before automatic transaction initiation (ATI) occurs.

CEMT INQUIRE|SET TERMINAL

CEMT INQUIRE|SET TERMINAL



Function

Note: If you are familiar with network names you may prefer the CEMT NETNAME command. See “CEMT INQUIRE|SET NETNAME” on page 340.

The INQUIRE TERMINAL command retrieves information about CICS terminal definitions in the CICS control region.

The SET TERMINAL command changes the CICS terminal definition in the CICS control region. It cannot be used for APPC sessions.

The status line shows the following information:

- The name of the CICS terminal, Ter (termid).
- The CICS transaction ID that is associated with the CICS terminal, Tra (tranid).
- Whether the CICS terminal is in service (Ins) or not (Out).
- Whether the CICS terminal has ATI ability (Ati).
- Whether the CICS terminal has TTI ability (Tti).
- The network name associated with the CICS terminal, Net (netname).
- Whether the CICS terminal is acquired (Acq) or not (Rel).

Required parameters

ACQUIRED

The CICS control region is in session with the CICS terminal.

ALL

All CICS terminals are checked.

ATI

The CICS terminal is to be available for use by CICS transactions that are initiated automatically from within CICS; or, if the CICS terminal is an ISC session, by CICS transactions that are using this session as an alternative facility to communicate with another system.

INSERVICE

The CICS terminal is to be available for use.

NETNAME(netname)

This is the name (1–8 characters) by which the CICS terminal is known to the network, as specified by the NETWORK parameter of the TCT.

NOATI

The CICS terminal is not available for use by CICS transactions that are initiated automatically from within CICS.

Note: NOATI and NOTTI cannot both be specified.

NOTTI

The CICS terminal is not available for use by CICS transactions that are initiated from this CICS terminal.

Note: NOTTI and NOATI cannot both be specified.

OUTSERVICE

The CICS terminal is not available for use. Setting a CICS terminal OUTSERVICE means that the CICS terminal can no longer be used by CICS transactions. If PURGE is also specified, any CICS transaction using the CICS terminal is terminated abnormally. If PURGE is not specified, the CICS transaction is allowed to terminate normally, but no further CICS transactions are allowed to use the CICS terminal. Setting a CICS terminal OUTSERVICE causes it to be released, either immediately or when the current CICS transaction has terminated.

PURGE

Any CICS transaction running with this CICS terminal is purged, only if system and data integrity can be maintained.

RELEASED

CICS is not in session with the CICS terminal. Setting a CICS terminal RELEASED causes the CICS shell to be terminated. Current CICS transactions are allowed to finish unless PURGE is also specified.

REMOTESYSTEM(sysid)

This is the name (1–4 characters) of the remote system associated with the CICS terminal as specified in the SYSID parameter of the TCS.

TERMINAL(termid)

This is the name (1–4 characters) of the CICS terminal as specified by the CICSDEV parameter of the TCT.

TRANSACTION(tranid)

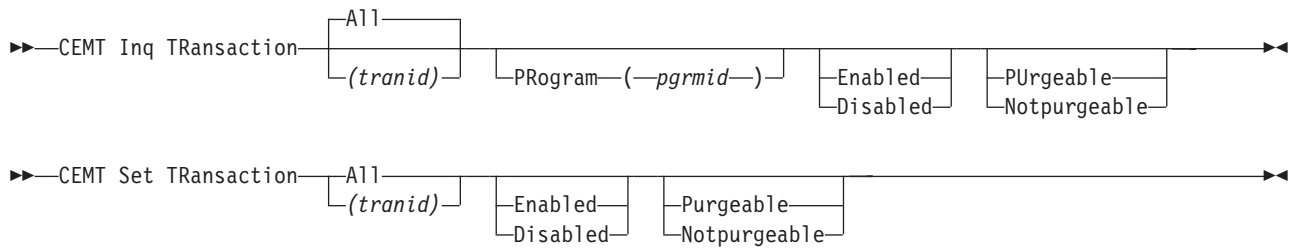
This is the name (1–4 characters) of the CICS transaction currently being processed with this CICS terminal.

TTI

The CICS terminal is to be available for use by CICS transactions that are initiated from this CICS terminal.

CEMT INQUIRE/SET TRANSACTION

CEMT INQUIRE/SET TRANSACTION



Function

The INQUIRE TRANSACTION command retrieves information about CICS transaction definitions in the CICS control region. You can use the DISCARD TRANSACTION function from an INQUIRE TRANSACTION screen.

The SET TRANSACTION command changes the transaction definition in the CICS control region.

The status line shows the following information:

- The name of the CICS transaction, Tran (trandid).
- The program name, Pro (pgrmid).
- Whether the CICS transaction is enabled (Ena) or disabled (Dis).
- Whether the CICS transaction can be purged (Pur) or not (a blank).

Required parameters

ALL

All CICS transactions are checked.

DISABLED

The CICS transaction is not available for use. CICS transactions that have identifiers beginning with "C" cannot be disabled because these are reserved for use by CICS.

ENABLED

The CICS transaction is available for use.

NOTPURGEABLE

The CICS transaction cannot be purged.

PROGRAM(pgrmid)

This is the name (1–8 characters) of the CICS program defined to process the CICS transaction, as specified by the PGMID parameter of the PCT.

PURGEABLE

The CICS transaction can be purged.

TRANSACTION(trandid)

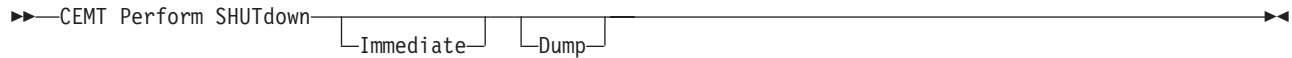
This is the name (1–4 characters) of the CICS transaction as specified by the TRANSID parameter of the PCT.

CEMT PERFORM commands

There are two CEMT PERFORM commands:

- PERFORM SHUTDOWN, to close a control region
- PERFORM SNAP, to take a control region dump

CEMT PERFORM SHUTDOWN



Function: The PERFORM SHUTDOWN command shuts down the CICS control region. If PERFORM SHUTDOWN is entered with no other options, all CICS tasks are allowed to finish. Terminal sessions are allowed to terminate normally, before the CICS control region is shut down.

Required parameters:

DUMP

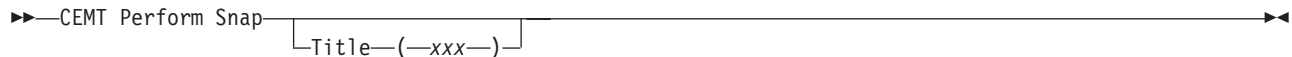
A CICS control region dump is produced before the completion of the termination process.

IMMEDIATE

The CICS control region is shut down immediately, terminating all active CICS tasks. This could cause CICS to abend if CICS tasks are still running.

CEMT PERFORM SNAP

CEMT PERFORM SNAP



Function: The PERFORM SNAP command produces a CICS control region dump. CICS/400 continues processing.

Required parameters:

TITLE(xxx)

A string of 1–32 characters. If the title includes spaces, the whole title must be enclosed within single quotation marks (' ').

Note: The **Title** option is accepted but ignored. On other CICS platforms, the title is appended to the beginning of the dump. On CICS/400, it has no effect. It is included here for compatibility with other CICS platforms.

The use of CEMT PERFORM SNAP prevents temporarily all other CICS tasks from running, so CICS users may experience delays in response.

Chapter 14. CRTE—CICS routing transaction

The CICS CRTE transaction is used to run CICS transactions that reside on a remote system.

The usual way of running a CICS transaction that resides on a remote system is to define the CICS transaction to CICS as remote. The installed CICS transaction definition names the sysid of the remote system. When the transid is entered for a CICS transaction that is defined as remote, CICS routes the request to the relevant system.

The CRTE CICS transaction provides another method of running a CICS transaction on a remote system. You can use the CRTE transaction, rather than defining a CICS transaction as remote, for infrequently used CICS transactions. CRTE is particularly useful for invoking the master terminal transaction, CEMT, on a particular system. You can invoke the CRTE transaction from any CICS terminal. However, the terminal through which CRTE is invoked must be defined on the remote system as a remote terminal, or be defined with SHIPPABLE(*YES) in the TCT entry.

The format of the routing transaction is:

►►—CRTE—SYSID—=—*sysid*——————►►

SYSID

This is the name (4 characters) of the CICS system on which the transaction is to run.

Note: Any parameters that are entered after the SYSID option are reported.

The CICS routing transaction verifies that the specified CICS system is known and is available. If it is, a message is displayed confirming that a routing session to the required CICS system has been started.

When this message is received, you can clear the screen and enter the CICS transid by which the required remote transaction is known on the remote system. In fact, you use the CICS terminal as if it were connected directly to the remote CICS system.

The CICS transactions that can be invoked include pseudoconversational CICS transactions, and the CRTE transaction itself. For information about pseudoconversational transactions, see the *CICS for iSeries Application Programming Guide*.

You cannot use PA or PF keys to invoke CICS transactions under the CRTE transaction.

You end a routing session by entering CANCEL. If you are signed on to a remote system, then the CANCEL automatically signs you off.

When a routing session has ended, the following message is displayed:

Routing session to system sysid ended

CRTE

Notes:

1. If the CRTE transaction was used to route CICS transactions through more than one system, CANCEL must be entered the same number of times as the CRTE transaction was entered to start the routing session.
2. While a CICS terminal is in a routing session with another system (that is, during the period between entering CRTE and CANCEL), the CICS terminal cannot receive messages delivered by automatic transaction initiation (ATI).
3. If you use CRTE to run CEDA remotely, you can use only the Install group function; you cannot work with a table. To process resource definitions, you need to interact with the OS/400 command interface, which you cannot do through CRTE.

Part 5. Appendixes

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
500 Columbus Avenue
Thornwood, NY 10594
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 106-0032, Japan

The following paragraph does not apply to 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 information 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.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

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 Corporation
Software Interoperability Coordinator
3605 Highway 52 N
Rochester, MN 55901-7829
U.S.A.

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 information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy,

modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This publication is intended to help you to configure and administer applications that run in the CICS for iSeries environment. This publication documents General-Use Programming Interface and Associated Guidance Information provided by CICS for iSeries.

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

General-Use Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following: **General-Use Programming Interface:**

Trademarks

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

AD/Cycle
Advanced 36
AIX
Application System/400
AS/400
CICS
CICS/ESA
CICS/VSE
DB2
e (logo)
IBM
IDPS
Intelligent Printer Data Stream
iSeries
Operating System/400
OS/2
OS/390
OS/400
System/36
System/38
TXSeries

VSE/ESA
z/OS
400

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

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

Glossary

This glossary defines special CICS terms used in this book; and words used differently from their everyday meaning.

If you can't find the term you are looking for, try the glossary in one of the books mentioned in the bibliography, page viii, or the *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994..

A

abend. An abnormal termination of CICS.

abend code. A four-character code indicating the type of CICS error that occurred.

access path. An alternative index or order to the file.

Advanced Program-to-Program Communication (APPC). The general term for LUTYPE6.2 protocol under Systems Network Architecture (SNA).

AID. See *automatic initiation descriptor*.

alternate index. An index based on an alternate key. It allows the file to be processed in a secondary key order.

APAR. See *Authorized Program Analysis Report*.

API. See *application programming interface*.

API commands. CICS commands supported for EXEC CICS statements. CICS/400 includes a subset of mainframe CICS supported commands.

API translator. See *translator*.

APPC. See *Advanced Program-to-Program Communication*.

application programming interface (API). The facility that allows application programmers to access CICS facilities by including EXEC CICS commands in their programs.

APPLID. In the SIT parameter, the 1 through 8 character application name of the CICS system. In the TCS parameter, the 1 through 8 character name by which this control region is known to other remote systems or regions.

ATI. See *automatic transaction initiation*

attention identifier (AID) keys. The enter key, the clear key, the PF keys, and the PA keys are all known as attention identifier (AID) keys. The enter key, and

the PF keys transmit data from the terminal's buffer to the system; the others just give an indication of which key was pressed.

Authorized Program Analysis Report (APAR). A report created to report a problem with an IBM licensed program.

autoinstall terminal. A method of defining terminals with a mask in the SIT, which results in a unique terminal identifier for the terminals in your system without having to do individual entries for each one.

automatic file closure. CICS/400 provides an automatic file closure facility based on an option selected in the SIT. The maximum number of open files is specified, and a time limit for a file to be open with no activity.

automatic initiation descriptor (AID). When an ICE expires for a timer-related task, it then becomes an AID. If all its required resources are available, it becomes an enabled AID; if it is waiting for a resource to become free, it is a suspended AID. Refer to *ICE*.

automatic transaction initiation (ATI). When data is sent to an intrapartition destination and the number of entries reaches a predefined trigger level, it can be specified that a transaction be started automatically to process the data in the intrapartition destination queue. A transaction can also start automatically another transaction at a specified terminal. See *control interval*.

auxiliary storage – TS queue. A temporary storage queue that is in a physical file managed by CICS. Auxiliary storage should be used to store large amounts of data, or data needed for a long period of time. Contrast with *main storage – TS queue*.

auxiliary trace. When this option is selected, trace entries are written to an external file. Any or all of these trace entries can then be printed to help you search for a problem. Contrast with *internal trace*.

B

basic mapping support (BMS). A facility that handles data stream input and output from a terminal. Its use provides device and format independence for application programs.

batch shell. A shell started to handle interval control timer requests. The batch shell is transparent to the user, each user's program runs under its own user shell. Contrast with *user shell*.

BMS. See *basic mapping support*.

BMS, minimum function. Support is provided for 3270 displays and printers only. Minimum BMS supports extended attributes and large screens. It does not support cumulative mapping, terminal operator paging, routing, or message switching.

bottleneck. A symptom which characterizes a performance problem. It can be due to a task failing to start, failing to continue after starting, or taking a long time to complete.

browse. Sequential reading of a file or temporary storage queue, beginning with a specified record.

C

CEBR. A supplied transaction that allows the user to browse temporary storage (TS) queues from a CICS user shell environment.

CECI. The command level-interpreter transaction. This supplied transaction allows application programmers to interactively syntax check and test their API commands before incorporating them into CICS application programs.

CECS. A supplied transaction that allows syntax checking of API commands.

CEDA. The resource definition transaction. A supplied transaction used to handle the manipulation of the CICS table definitions. There are two major functions within CEDA: PROCESS and INSTALL. See separate entries for details.

CEDF. The transaction used to start the Execution Diagnostic Facility. See *Execution Diagnostic Facility*.

CEMT. The master terminal command. This supplied transaction allows the system administrator to inquire about or change the status of resources (including programs, transactions, files, queues, and terminals), and to shut down the control region.

CESE. Supplied transaction to sign off from CICS. This transaction signs the user off from the user shell.

CICS group. An OS/400 library containing the CICS resource definition for CICS tables.

CICS-value data areas (CVDA). CICS supplied values to certain data options on EXEC CICS commands.

CL. See *control language*.

client/server. The model of interaction in distributed data processing in which a program at one site sends a request to a program at another site and awaits a response. The requesting program is called a client; the answering program is called a server.

COBOL/400. The AD/Cycle COBOL compiler available for OS/400. One of the two supported languages for CICS/400 application development.

command. In an OS/400 environment usually refers to a CL command. Each CL command corresponds to a specific operation. Use of a CL command is usually quicker than the corresponding menu selections. In CICS, an instruction similar in format to a high-level language instruction; the statement begins with EXEC CICS.

cold start. One of the ways in which temporary storage and transient data queues are recovered when a CICS control region is started. A cold start indicates that these resources are cleared. Contrast with *warm start* and *emergency start*.

commit. Changes made to files are written or committed at a syncpoint.

commitment control. A means of grouping database file operations that allows the processing of a group of database changes as a single unit through the commit command, or the removal of a group of database changes as a single unit through the rollback command.

common work area (CWA). A work area that can be accessed by any transaction in the CICS system.

control block. A specialized storage area in shared system storage that is used by CICS to pass information between service modules.

control language (CL). The primary interface to the OS/400 operating system. Each command name refers to a command processing program in the system that performs the actions indicated by the command.

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

control region. The control region provides the control, scheduling, and work management mechanisms necessary to coordinate all the shared resources in CICS/400. A control region is started using the STRCICS command and ended using the ENDCICS command.

conversion vector table (CVT). This table contains entries that identify how data is to be converted when transporting data to or from a remote system.

CR. See *control region*.

CRTE. A supplied transaction used for routing transactions to another CICS system.

CSMT. A transient data queue that is a destination for messages. In CICS/400, it is up to the system administrator whether or not to define this queue.

CSMT log. A transient data destination used by CICS for writing terminal error and abend messages. The CSMT log should be defined in the DCT.

CVDA. See *CICS-value data areas*.

CVT. See *conversion vector table*.

CWA. See *common work area*.

D

database administrator. The person responsible for the design, development, integrity, and maintenance of databases.

data conversion. An intersystem routine used by function shipping to convert data from or before sending data to remote CICS sites. For example, an ASCII to EBCDIC conversion may be needed.

data conversion table. See *conversion template table*.

data description specification (DDS). A description of the user's database or device files that is entered into the system in a fixed form. The description is then used to create files.

data queue. A record oriented, first-in first-out, queue. In CICS/400 a data queue is created for each job. Data queues are the primary means used to notify the control region that it has work to process.

data queue waits. This wait occurs when resources are unavailable within either the control region or one of the shells.

DCT. See *destination control table*.

DDS. See *data description specification*.

deadlock. A contention for resources, where two programs are attempting to use the same resource (for example, update the same record) at the same time.

deferred work element (DWE). The deferred work element is the catalyst used to invoke "event driven" services controlled within CICS. DWE's cause a unit of work to be scheduled later, normally either at task termination or just before or just after syncpoint.

dequeue. CICS/400 release a resource held for exclusive use. On the OS/400, this term also means to "read" a data queue.

destination control table (DCT). A table describing each of the transient data destinations used in CICS. This table contains an entry for each extrapartition, intrapartition, and indirect destination.

distributed program link (DPL). This enables an application program running on one CICS system to link to another application program running in another CICS system.

distributed transaction processing (DTP). This process enables a CICS transaction to communicate with a transaction running in another system.

DPL. See *distributed program link*.

DTP. See *distributed transaction processing*.

dump. A snapshot of what was happening in CICS at the time the dump was taken, whether as a result of an abend or in response to a user request.

dump control. This facility handles program controlled dumping of the application program. In CICS/400 also referred to as *serviceability and dump*.

DWE. See *deferred work element*.

E

EDF. See *Execution Diagnostic Facility*.

EIB. See *Execution Interface Block*.

emergency start. One of the ways in which temporary storage and transient data queues are recovered when a CICS control region is started following an abnormal shutdown of the region. An emergency start affects these resources as defined in the System Initialization Table. This may result in one or more queues being cleared or recovered, that is, returned to their state prior to shutdown. Contrast with *cold start* and *warm start*.

emulation. An imitation of one computer or product by another so that both accept the same data, and achieve similar results.

enqueue. A CICS/400 user-defined resource is held for exclusive use. On the OS/400 this term also means to "write" a record to a data queue.

error message. On the OS/400 error messages (abends) from unsuccessful routines are displayed at the terminal, and additional help is available by using the PF1 key.

ESDS – entry sequenced data set. One of the file organizations supported to emulate VSAM. On an ESDS file each record is identified by its relative byte address (RBA). Records are held in the order in which they were first entered, with new records added at the end.

EXEC (EXECUTE) statement. An instruction similar in format to a high-level language instruction. It begins with EXEC CICS, then lists the command and options, and ends with END-EXEC.

Execution Diagnostic Facility (EDF). A facility that helps the application programmer to debug an application by stepping through its CICS commands. The programmer can change values in the application while it is running. To start EDF, you need to use the CEDF transaction.

EXEC Interface Block (EIB). A control block associated with a CICS task, this block is used for direct communication between CICS and command-level application programs. Several fields in the EIB, such as the RESP and RESP2 fields are often checked by the programmer to determine whether a CICS command was executed as expected.

extrapartition destination. A type of transient data queue. Extrapartition destinations can be accessed either within the CICS environment or outside of CICS/400; they can be defined as either input or output.

F

FCT. See *file control table*.

FFDC. See *First Failure Data Capture*.

file control. The facility for managing basic operations against a file (ADD, READ, DELETE, REWRITE, and BROWSE).

file control table. A table containing the characteristics of files accessed by file control.

First Failure Data Capture (FFDC). An OS/400 facility called when a serious error occurs within CICS. Information relating to the transaction being executed at the time may be found, along with some control region information.

function shipping. The process whereby CICS accesses resources when those resources are actually held on another CICS system.

G

GLT. See *group list table*.

group. A collection of CICS resource definitions (programs, BMS map sets, and table entries), that can be usefully exported together. A group normally includes all the elements for an application.

group list table (GLT). This table identifies the library and file names for resource mapping, that are to be installed when the control region is started.

I

ICE. See *interval control element*.

ideographic language. A language that uses pictures or symbols to represent a thing or an idea but not a particular word or phrase for it. Each picture or symbol, known as an ideogram, is supposed to suggest the idea or object rather than being the object itself. The languages that use this method of writing are Japanese, Korean, Traditional Chinese, and Simplified Chinese.

ILE. See *integrated language environment*.

ILE C. An IBM licensed program that is a Systems Application Architecture* (SAA*) platform C programming language. The ILE C program uses the ILE model on the iSeries system.

incorrect output. Data was missing, was corrupted, or was not formatted correctly.

indirect destination. A type of transient data destination that points to another destination within the DCT. See *destination control table*.

initialization. The preparation of a system, device, or program for operation. An operating system is initialized on startup or after a system failure.

initialization stall. A wait that occurs during initialization when a CICS system appears to be running normally but is not actually progressing through the various stages of initialization.

INSTALL table. One of the CEDA transaction functions. This function allows you to transfer CICS resource definitions in the specified group to the control region specified.

integrated language environment (ILE). A set of constructs and interfaces that provides a common run-time environment and run-time bindable application program interfaces for all ILE-conforming high-level languages.

internal tracing. An option whereby trace entries are written to an internal control region table. The table, which can be specified to wrap when full, is most appropriate if you do not need to capture a large number of trace entries. Contrast with *auxiliary trace*.

Intersystem Communication (ISC). This facility provides inbound and outbound support for communication from other computer systems.

interval control. The primary task of this facility is the handling, synchronization, and initiation of tasks requested by user-application programs and CICS internal service routines. Other functions include obtaining the formatted time for the user.

Interval control element (ICE). An entry under interval control that is waiting in an unexpired state. Its defined date and time (to become current) is still in the future. When an ICE expires it becomes an AID. Refer to *AID*.

intrapartition destination. A type of transient data queue used subsequently as input data to another task within CICS.

ISC. See *Intersystem Communication*.

IVP. See installation verification procedure.

installation verification procedure. This provides a sample online and maintenance application to verify a successful installation of CICS/400.

J

JCT. See *journal control table*.

job log. This message queue is attached to a particular job. This means that there is a separate job log for the control region and for each shell associated with that control region.

journal control. Provides the CICS user with the ability to write CICS journal records when required by the application for auditing purposes.

journal control table (JCT). The JCT describes the CICS user journals along with their access characteristics.

K

KSDS – key-sequenced data set. One of the types of file organization supported to emulate VSAM. Each record in the file is identified by a key within a predefined position of the record. Each key must be unique.

L

limited capability. The use of certain OS/400 commands can be restricted by setting a user's profile to limited capability.

local. Pertaining to the system, program, or device being described, as opposed to other systems, programs, or devices which are on other computer systems. Contrast with *remote*.

local space object. Located in the OS/400 system domain storage, it is used to maintain and keep track of CICS storage elements in both system and user storage objects.

logical file. A description of how data is to be presented to or received from a program. This description contains *no* data, but it defines record formats for one or more physical files. Contrast with *physical file*.

logical unit (LU). A port through which a user gains access to the services of networks.

logical unit of work (LUW). The work processed between syncpoints. In CICS/400 an implicit syncpoint occurs at normal task completion.

loop. Repeated execution of a portion of code.

LU. See *logical unit*.

LUW. See *logical unit of work*.

M

main storage – TS queue. A dynamic storage area managed by CICS under the temporary storage facility. Data in main storage does not survive from one CICS run to the next. Contrast with *auxiliary storage–TS queue*.

map. Screen data defined as fields using BMS macros which are generated into two forms, a physical map for display on part or all of a device, and a symbolic map referring to just the named input and output fields that are needed by the program processing the screen data. Maps are grouped together in a *map set*.

map set. A group of one or more (normally related) BMS maps.

master terminal operator. The person, usually the system administrator, responsible for monitoring and controlling resources in a CICS control region.

message file. The file holding the text of all CICS messages.

message queue. A queue to which error and informative messages get directed. In CICS/400 the major message queues are the QSYSOPR queue, the job log, and the QHST log file.

multithread test. This type of test involves several concurrently active transactions. Whether the new function can coexist with other related functions is tested. Contrast with *single-thread test*.

N

national language support (NLS). A feature that allows the user to communicate with the system in the national language chosen by the user.

network. The hardware equipment (for example, terminals and lines) that supplies the connections between terminals and hosts or other systems.

nonrecoverable (requests). Requests on a queue or in a file that are lost in the event of transaction or system failure. Contrast with *recoverable*.

nonshared storage. Storage areas created for and used exclusively by individual user shells.

null value. A value returned on an INQUIRE or SET command when a requested attribute is not applicable or not available. Its value depends on the format of the data-area set up to receive the attribute.

O

object. A named storage space that consists of a set of characteristics that describes itself. An object is anything that exists in and occupies space in storage and on which operations can be performed. Some examples of objects are programs, files, libraries, and folders.

object oriented system. On the OS/400 everything is treated as an object. That is, files, programs, user spaces, and data queues are all treated as objects.

open data path (ODP). A path created when a file is opened. An ODP contains information about the merged file attributes and information returned by input or output operations. The ODP only exists while the file is opened.

Operating System/400 (OS/400). Pertaining to the IBM licensed program that can be used as the operating system for the iSeries system.

P

panel. The screen layout of such items as the initial cursor position, entry fields, selection fields, and list fields.

PCT. See *program control table*.

PF keys. Program function keys used within CICS and on the OS/400 to perform certain preprogrammed functions. For example, PF3 is normally set up to take you back to a previous screen.

physical file. A description of how data is to be presented to or received from a program and how data is actually stored in the database. A physical file contains one record format and one or more members. Contrast with *logical file*.

physical map. See *map*.

PPT. See *processing program table*.

precompiler. A program that manages preparation of source code for compilation. Also see *translator*. Often used as synonyms.

printer spooling. This CICS/400 facility provides support for writing data to OS/400 print spools. Only *printed* output is supported by CICS/400.

problem determination. The process you use to determine where you have a problem with your CICS system.

problem symptom. Unexpected results from a CICS system, an error condition, or error messages.

processing program table (PPT). A table defining the application programs and BMS maps that can be run under CICS.

PROCESS table. A CEDA transaction function used to change and maintain CICS resource definitions.

program check. This indicates that an error in a program has caused CICS transaction to terminate abnormally under the control of CICS which issues appropriate messages.

program control. This facility handles the flow of control among application programs.

program control table (PCT). A table defining the transactions that can be processed by the system. Each transaction id is paired with the name of the program that CICS executes when the transaction is invoked.

program temporary fix (PTF). A change to CICS between releases.

protocol, data link. A set of rules for data communication over a data link; in terms of a transmission code, a transmission mode, and control and recovery procedures.

PTF. See *program temporary fix*.

Q

QHST log file. A message destination that is an alternative to the job log.

QSYSOPR message queue. The system operator's queue. Few messages get written to this queue, but it should always be checked if there are serious errors within CICS.

queue. A line or list formed by items in a system waiting for service; for example, tasks to be performed, or messages to be transmitted.

R

RETAIN. IBM maintained database of all known problems with a particular release of CICS.

RDO. See *resource definition online*.

recoverable (requests). Requests on a queue or in a file that can be restored to their previous state at the exit of the last task or syncpoint after a transaction or system failure.

reentrant code. When a program is reentrant it is serially reusable. Each time you enter the program a

fresh copy of working storage is provided. If any values need to be saved, you must save them in other storage areas or files.

regression test. A complete test of the existing system along with any changes.

remote. Pertaining to a system, program, or device that is accessed through a telecommunication line. Contrast with *local*.

resource definition online (RDO). This facility lets you define certain CICS resources interactively while CICS is running. Specifically RDO lets you define terminals, programs, and transactions interactively. Its use is normally restricted to the system administrator.

resource management. The facility that keeps track of what system resources are being used by mapping the CICS identification name to the underlying system resources.

resource tables. Related types of resource information are stored within CICS in tables or control blocks. For example: CVT, PPT, SIT, or TCT.

rollback. The process of canceling changes made by a transaction to recoverable resources, following the failure of that transaction.

routing transaction. A supplied transaction (CRTE) that allows an operation at a terminal owned by one CICS system to sign on to another CICS system connected by an APPC link.

RRDS – relative record data set. One of the types of file organization supported to emulate VSAM. This file organization contains fixed-length records identified by a relative record number (RRN). When you add a new record you can either specify the RRN or let VSAM assign the next sequential number.

runaway task. A transaction that features instructions that are repeated continually without issuing a condition to end the loop.

S

security. A mechanism to ensure resources are protected from unauthorized access.

serviceability and dump. See *dump control*.

shared storage. Storage shared between the control region and other processes.

shell (SH). The CICS facility that provides the work management mechanism to build and refresh the application programming environment needed to run CICS transactions. A shell is started using the STRCICSUSR command and ended using the ENDCICSUSR command. Also see *user shell*.

shutdown. The process of ending a CICS control region in a controlled way by using the CEMT transaction or as a result of a system failure.

space objects. System objects that manage storage requests for internal CICS and user application requirements, that is, local space, system space and user space.

single-thread test. Test of a single application or transaction running by itself. Contrast with *multithread test*.

SIT. See *system initialization table*.

SNA. See *System Network Architecture*.

SQL. See *structured query language*.

stop condition. A specified condition for stopping the execution of a transaction which is being debugged by the Execution Diagnostic Facility (EDF).

storage control. This facility controls requests for main storage to provide intermediate work areas not automatically provided by CICS. The use of these areas is requested with a GETMAIN command.

storage violation. Storage has somehow become corrupted. Part of CICS may have been overwritten by another program.

structured query language. A language that is used to access data in a relational database by using simple conversational type statements.

supplied transactions. General-purpose transactions supplied by CICS/400 to perform general CICS functions required by many users, such as debugging (CEDF).

switchable file. In the CICS/400 journal control facility, a new CICS journal file is created and opened when an out-of-space condition occurs during a CICS journal write operation.

symbolic map. The set of input and output fields of a BMS map which are referenced by a program, held in a COBOL copybook or C header file for inclusion in the program.

syncpoint. The commitment or backout of OS/400 recoverable resources within a task's logical unit of work. A syncpoint may be either explicitly requested, or implicitly initiated at task termination.

system administrator. Any person authorized to use or modify resources or use the CEMT transaction.

system dump. A dump of the whole control region.

system initialization table (SIT). A table containing parameters used to start the control region's system initialization process.

system programming commands. System programming commands are used for monitoring and changing certain parameters within the control region.

systems information access. This facility provides the OS/400 system administrator and application developers with the ability to inquire on and modify certain CICS runtime resource definitions.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of networks.

system storage. These storage areas are used specifically for internal CICS requirements. Contrast with *user storage*.

T

tables. Resources used in the CICS/400 system are stored in resource tables and used to control the interaction between the different components in the system, for example programs and terminals.

task. The execution of one or more application programs for a *specific user*. Several users may invoke the same transaction, but each execution of that transaction is treated as a separate task.

task control. This facility controls resources at task syncpoint, and at normal or abnormal termination.

TCT. See *terminal control table*.

TCTUA. See *terminal control table user area*.

temporary storage (TS). The facility that allows application programs to store data in a temporary storage queue for later retrieval.

TD. See *transient data*.

temporary storage table (TST). A table describing temporary storage queues. For temporary storage data to be recoverable by CICS, if the system terminates abnormally, data identifiers for that temporary storage queue must be specified in the TST.

terminal. In CICS, a device equipped with a keyboard and a display; capable of sending and receiving information over a communication channel.

terminal control. This facility handles addressing, and transmission error detection and correction for terminals (both display and printers) associated with the local CICS system. It also handles the intercommunication data queue used by CICS.

terminal control system entry (TCS). A table defining the connections between CICS systems.

terminal control table (TCT). A table describing the terminals and logical units within a CICS network.

terminal control table user area (TCTUA). An area used to pass information between application programs, but only if the same terminal is associated with the application programs involved.

timer-related event. An interval control function that is used to support events that are delayed, suspended, or restarted after a time interval.

trace. A debugging aid for system and applications programmers. This facility produces trace entries at set system points or in response to trace commands.

transaction. A transaction is identified to CICS by a four-character code called a transid. It is often invoked when an operator enters a transid in column 1 on line 1 of a terminal.

transaction abend. All transactions are identified by a four-character code called a transid. If there is a problem and a transaction abends, you get an error message that identifies the transaction and the program. You also get a dump of the transaction.

transaction dump. A transaction dump consists of a formatted dump for the COBOL program active at the time the dump was requested. It indicates where the error occurred within the COBOL program.

transaction routing. The facility that provides support for inbound and outbound terminal requests from another CICS system connected by an APPC link.

transaction work area (TWA). A work area that exists only for the duration of a transaction. Consequently, you can use it to pass data among programs executed in the same transaction, but not between transactions.

transient data (TD). This facility provides the user with the ability to read and write data in sequential queues.

translator. A program that converts (translates) EXEC CICS commands found in source code into host language statements, including variable assignments and a call to the CICS EXEC interface program. Also see *precompiler*.

trigger level. The number of requests for output to a transient data queue that must be reached before automatic transaction initiation occurs.

TS. See *temporary storage*.

TST. See *temporary storage table*.

TWA. See *transaction work area*.

two-phase commit. The protocol that permits updates to protected resources to be committed or rolled back as a unit. During the first phase, partners in a

conversation are asked if they are ready to commit. If all partners respond positively, they are asked to commit their updates. Otherwise, they are asked to roll back their updates.

U

user-based pricing. A pricing option that provides the capability for the customer to pay for a licensed program on the basis of the number of users.

user liaison. The department responsible for communication between data processing functions and end users.

user shell. An interactive shell initiated by a user by issuing a STRCICSUSR CL command. The user's CICS application program runs directly under this CICS facility. Also see *shell*.

user storage space. These storage areas are obtained and managed by CICS, but are used by application modules requesting main storage. Contrast with *system storage*.

user tracing. User tracing allows for the recording of user-defined trace areas. User tracing can be used with either internal or auxiliary tracing.

V

VSAM – Virtual Storage Access Method. In CICS support, access to all three types of emulated VSAM files is provided: KSDS, ESDS, and RRDS. Refer to separate types for more detailed information.

VTAM. Virtual Telecommunications Access Method

W

wait. A state in which a task's execution has been suspended; for example, waiting for a resource to become available.

warm start. One of the ways in which temporary storage and transient data queues are recovered when a CICS control region is started following a normal shutdown of the region. A warm start affects these resources as defined in the System Initialization Table. This may result in one or more queues being cleared or recovered, that is, returned to their state prior to shutdown. Contrast with *cold start* and *emergency start*.

Index

Special Characters

? and CEMT 328
* and CEMT 327
*ENDJOB 124
+ in CEMT syntax 327
+ in scrolling under CEMT 328

A

ACCMTH parameter
 ADDCICSFCT 181
 CHGCICSFCT 187
ACCPH parameter
 ADDCICSFCT 181
 CHGCICSFCT 187
ACQUIRED option
 CEMT INQUIRE|SET
 CONNECTION 335
 CEMT INQUIRE|SET
 TERMINAL 346
acquiring VTAM terminals 346
ADDABLE option
 CEMT INQUIRE|SET FILE 337
ADDCICSCVT command 143
ADDCICSDCT command 161
ADDCICSFCT command 177
ADDCISGLT 48
ADDCISGLT command 194
ADDCICSJCT command 202
ADDCICSPCT command 214
ADDCISPPPT command 227
ADDCICSSIT command 62, 89, 239
 SHRSTG parameter 68, 72
ADDCICSTCS command 261
ADDCICSTCT command 275
ADDCICSTST command 296
adding resource definitions
 CVT 143
 DCT 161
 FCT 177
 GLT 194
 JCT 202
 PCT 214
 PPT 227
 SIT 239
 TCS 261
 TCT 275
 TST 296
ADDLIBLE command 24
adoptive authority
 CRTICISC CL command 78
 CRTICISCBL CL command 78
 CRTCLPGM CL command 78
AEGTCACP (autoinstall control
 program) 309
AEGTCACP (autoinstall user-replaceable
 program)
 sample control program 319
 suggestions for use 319

AEGTCACP (CICS-supplied
 program) 312
ALL operand
 general statement 326
ALL option
 CEMT INQUIRE|SET
 AUXTRACE 334
 CEMT INQUIRE|SET
 CONNECTION 335
 CEMT INQUIRE|SET FILE 337
 CEMT INQUIRE|SET
 JOURNALNUM 339
 CEMT INQUIRE|SET
 PROGRAM 341
 CEMT INQUIRE|SET TASK 343
 CEMT INQUIRE|SET
 TDQUEUE 345
 CEMT INQUIRE|SET
 TERMINAL 346
 CEMT TRANSACTION 348
alternate index 89
ALTSCN parameter
 ADDCICSTCT 281
 CHGCICSTCT 289
ALTSUFFIX parameter
 ADDCICSTCT 279
 CHGCICSTCT 287
APISET parameter
 ADDCISPPPT 229
 CHGCISPPPT 233
APPLID parameter
 ADDCICSSIT 241
 CHGCICSSIT 250
ASCII data 142
ATI (automatic transaction
 initiation) 345
ATI option
 CEMT INQUIRE|SET
 TERMINAL 346
ATISTS parameter
 ADDCICSTCT 279
 CHGCICSTCT 287
AUTINSTMODEL option
 CEMT DISCARD 333
AUTO option
 CEMT INQUIRE|SET
 JOURNALNUM 339
autoinstall control program 309
 batch shell 311
 CATD 311
 deleting 311
 program object security 311
 security 311
 storage 311
 using CEMT INQ/SET 312
 using EXEC CICS
 INQUIRE|SET 312
autoinstall copybook 312
autoinstall terminals 12, 96, 307, 308
 DEVCTL 242
 DEVCTL parameter 308

autoinstall terminals (*continued*)
 DEVMODEL 278
 DEVMODEL parameter 305
 TCT entries 308
 unique termids 308
autoinstall_model_name option
 CEMT INQUIRE|SET
 AUTINSTMODEL 334
automatic control region definition 34
automatic file closure 89
automatic installation of terminals
 COMMAREA at logon 313
 control program
 action at delete 317
 action at install 313
 action on return 317
 information returned to
 CICS/400 315
 naming 318
 testing and debugging 318
 sample autoinstall control program
 AEGTCACP 309
 the sample programs
 customizing 319
automatic resource definition 34
automatic transaction initiation
 (ATI) 345
auxiliary trace
 CEMT INQUIRE|SET 334
AUXTRACE format
 CEMT transaction 334
AUXTRCCTL parameter
 ADDCICSSIT 243
 CHGCICSSIT 252

B

basic mapping support (BMS) 7
BROWSE option
 CEMT INQUIRE|SET FILE 337

C

C option
 CEMT INQUIRE|SET
 PROGRAM 341
CATD 311
CATD transaction 65
CCIN transaction 65
CDEPAGE parameter
 ADDCICSTCS 262
 CHGCICSTCS 267
CEBR transaction 65
CECI transaction 65
CECS transaction 65
CEDA transaction 51, 58, 65
 F key functions 58
CEDF option
 CEMT INQUIRE|SET
 PROGRAM 341

CEDF transaction 65
 CEMT
 SET FILE CLOSED 89
 CEMT INQUIRE AUTOINSTALL 318
 CEMT SET AUTOINSTALL 318
 CEMT transaction
 ? character 328
 * symbol 327
 + in scrolling 328
 AUTINSTMODEL 333
 AUXTRACE 334
 blank fields in a display 328
 CEMT on the command line 325
 CONNECTION 335
 DISCARD 332
 FILE 336
 INQUIRE 325
 INTRACE 338
 JOURNALNUM 339
 NETNAME 340
 overtyping INQUIRE displays 331
 PROGRAM 340
 program function (PF) keys 330
 question mark preceding CEMT 328
 request formats 325
 scrolling a display 328
 SET 325
 SHUTDOWN 349
 SNAP (dump) 349
 SYSTEM 342
 tab key 330
 TASK 343
 TDQUEUE 344
 TERMINAL 346
 TRANSACTION 348
 CESF transaction 65, 129
 changing a group description 137
 changing resource definitions
 CVT 149
 DCT 167
 FCT 183
 JCT 206
 PCT 218
 PPT 231
 SIT 248
 TCS 265
 TCT 282
 TST 298
 CHGCICSCVT command 149
 CHGCICSDCT command 167
 CHGCICSFCT command 183
 CHGCICSGRP command 46, 137
 CHGCICSJCT command 206
 CHGCICSPCT command 218
 CHGCICSPPT command 231
 CHGCICSSIT command 248
 SHRSTG parameter 68, 72
 CHGCICSTCS command 265
 CHGCICSTCT command 282
 CHGCICSTST command 298
 CICS application
 security considerations 75
 CICS clients, server support for 5
 CICS groups
 group list table 47, 193
 multiple 46
 CICS-supplied program
 for automatic installation of terminals
 (AEGTCACP) 312
 for IVP 23
 to define a control region 34
 CICS/400
 restrictions 16
 CICS/400 architecture 5
 CICS/400 user interfaces
 dump 16
 resource definition 12
 source translator 15
 SQL 10
 trace 16
 CICSDEBUG parameter
 ADDCICSPPT 229
 CHGCICSPPT 233
 CICSDEV parameter
 ADDCICSDCT 164
 ADDCICSTCT 277
 CHGCICSDCT 170
 CHGCICSTCT 285
 DSPICSTCT 291
 RMVCICSTCT 293
 WRKCICSTCT 294
 CICSMAP parameter
 ADDCICSPPT 228
 CHGCICSPPT 232
 CL commands 7
 CL commands, entering 52
 client terminals, autoinstall 305
 clients, server support for 5
 CLOSED option
 CEMT INQUIRE|SET
 JOURNALNUM 339
 CEMT INQUIRE|SET
 TDQUEUE 345
 CLRGROUP parameter 32
 CRTICISGRP 136
 CMDCICS CL command 7
 CMDTYPE parameter
 ADDCICSCVT 144
 CHGCICSCVT 150
 DSPICSCVT 156
 RMVCICSCVT 158
 WRKCICSCVT 160
 CMPX transaction 65
 CNVCHRID parameter
 ADDCICSCVT 145
 CHGCICSCVT 151
 CNVINP parameter
 ADDCICSCVT 145
 CHGCICSCVT 151
 COBOL option
 CEMT INQUIRE|SET
 PROGRAM 341
 cold start 113, 115
 command program objects
 resource checking 76
 commands, CEMT
 DISCARD 332
 INQUIRE and SET 333
 PERFORM 348
 commitment control 6, 104, 105
 commitment control requirements 104
 compression, datastream 274
 CONNECTION(sysid)
 CEMT INQUIRE|SET
 CONNECTION 335
 control region
 AEGSAMPLSO object 69
 AEGSAMPYSYS object 71
 AEGSAMPUSR object 71
 controlled shutdown 111
 data queue 108
 data queue messages 109
 deferred work element (DWE) 110
 extrapartition transient data 86
 initialization and termination
 table 108, 111
 internal storage 70
 ISC processing 111
 priority 67
 shared resources 5, 107, 110
 special features 110
 storage requirements 68
 subsystem processing 112
 supplied 33
 work flow 107
 runtime processing 109
 shutdown 111
 work management 5, 107
 control region objects 110
 conversion information 143, 145, 151
 conversion vector table
 adding an entry 143
 changing an entry 149
 CNVINP parameter 143, 145, 151
 conversion information 143
 displaying an entry 155
 key conversion 142
 KEYINF parameter 142, 146, 152
 removing an entry 157
 selection criteria 143
 SLICTL parameter 143, 147, 153
 working with entries 159
 CONVERT parameter
 STRCICS 49, 50, 107, 113
 converting resource definitions 49, 50
 CPMI transaction 65
 CPYLIB command 24
 create CICS group command 32
 create journal command 28
 create journal receiver command 28
 creating a group 135
 creating the CL program 28
 CRSR transaction 65
 CRTICSCBL command 76
 autoinstall control program 310
 IVP 26
 CRTICISGRP command 46, 135
 IVP 32
 CRTICISMAP command 25
 CRTCLPGM command 28
 CRTDUPOBJ command 27
 CRTE transaction 65, 351
 CRTJRN command 28
 CRTJRNRCV command 28
 CSMT transaction 65
 CSMT log
 defining 87, 163
 for autoinstall 311
 CSRR transaction 65

CSSF transaction 65
CTIN transaction 65
CTLRGN parameter 113, 119, 126, 128
INSCICSGRP 139
CVMI transaction 65

D

data conversion 142
data management 82
DATA parameter 126
DATASET option
CEMT INQUIRE|SET
JOURNALNUM 339
datastream compression 274
DATFORM parameter
ADDCICSSIT 241
CHGCICSSIT 251
DEFICISRGV CL command 34
DELAY parameter 119, 129
DELETE option
CEMT INQUIRE|SET FILE 337
deleting a group 138
deleting autodefinitions 38
DEST option
CEMT INQUIRE|SET TASK 343
DEST parameter
ADDCICSDCT 162
CHGCICSDCT 169
DSPCICSDCT 174
RMVCICSDCT 175
WRKCICSDCT 177
destination control table
adding an entry 161
changing an entry 167
displaying an entry 173
removing an entry 175
working with entries 176
destinations for data
extrapartition 85
intrapartition 85
DESTSTS parameter
ADDCICSDCT 163
CHGCICSDCT 169
DEVACQ parameter
ADDCICSTCT 281
CHGCICSTCT 290
DEVCHRID parameter
ADDCICSTCT 280
CHGCICSTCT 288
DEVCTL parameter 308
ADDCICSSIT 242
CHGCICSSIT 251
DEVD parameter
ADDCICSTCT 278
CHGCICSTCT 286
DEVMODEL parameter 305
ADDCICSTCT 278
CHGCICSTCT 286
DEVRCYACN parameter 124
DEVSTS parameter
ADDCICSTCT 279
CHGCICSTCT 287
DEVTYPE parameter
ADDCICSDCT 164
ADDCICSTCT 278
CHGCICSDCT 170

DEVTYPE parameter (*continued*)
CHGCICSTCT 285
DFHTCUDO 312
DISABLED option
CEMT INQUIRE|SET FILE 337
CEMT INQUIRE|SET
PROGRAM 341
CEMT INQUIRE|SET
TDQUEUE 345
CEMT TRANSACTION 348
DISCARD, CEMT
AUTINSTMODEL 333
FILE 333
PROGRAM 333
TRANSACTION 333
display station security 75
displaying resource definitions
CVT 155
DCT 173
FCT 189
GLT 196
JCT 209
PCT 222
PPT 235
SIT 258
TCS 269
TCT 291
TST 300
DLTICISGRP command 46, 138
DPLSUBSET option
CEMT INQUIRE|SET
PROGRAM 341
DSCOMP parameter
ADDCICSSIT 247
ADDCICSTCT 282
CHGCICSSIT 257
CHGCICSTCT 290
DSNAME option
CEMT INQUIRE|SET FILE 337
DSPCICSCVT command 155
DSPCICSDCT command 173
DSPCICSFCT command 189
DSPCICSGLT command 196
DSPCICSJCT command 209
DSPCICSPCT command 222
DSPCICSPPT command 235
DSPCICSSIT command 258
DSPCICSTCS command 269
DSPCICSTCT command 291
DSPCICSTST command 300
DUMP option
CEMT PERFORM SHUTDOWN 349
DUMP parameter
ADDCICSPCT 216
ADDCICSSIT 242
CHGCICSPCT 221
CHGCICSSIT 251
dynamic installation, resource
definition 58
dynamic installation, resource
definitions 48, 49, 139

E

EBCDIC data 142
EDTOBJAUT command 78
emergency start 113, 115

EMPTYREQ option
CEMT INQUIRE|SET FILE 337
ENABLED option
CEMT INQUIRE|SET FILE 337
CEMT INQUIRE|SET
PROGRAM 341
CEMT INQUIRE|SET
TDQUEUE 345
CEMT TRANSACTION 348
ENDCICS command 76, 118
ENDCICSUSR command 76, 127
EXEC CICS INQUIRE command
for autoinstall 318
EXEC CICS SET command
for autoinstall 318
EXTRA option
CEMT INQUIRE|SET
TDQUEUE 345
extrapartition destination 85
extrapartition storage queues 43
groups 44
extrapartition transient data 86

F

FACILITY option
CEMT INQUIRE|SET TASK 343
fields, blank, in a display 328
file closure
automatic 89
CEMT SET FILE CLOSED 89
SET FILE CLOSED 89
file control 6
file control table
adding an entry 177
changing an entry 183
displaying an entry 189
removing an entry 191
working with entries 192
file objects
CEMT requests 336
FILE option
CEMT DISCARD 333
FILE parameter
ADDCICSDCT 164
ADDCICSFCT 180
CHGCICSDCT 171
CHGCICSFCT 186
file types, supported 10, 82
FILECTL parameter
ADDCICSSIT 242
CHGCICSSIT 252
FILECTL parameter example 89
FILEID parameter
ADDCICSFCT 179
CHGCICSFCT 185
DSPCICSFCT 190
RMVCICSFCT 191
WRKCICSFCT 193
FILENAME option
CEMT INQUIRE|SET FILE 337
files
automatic file closure 89
recoverable 90, 104
FILESTS parameter
ADDCICSFCT 181
CHGCICSFCT 186

FULLAPI option
CEMT INQUIRE|SET
PROGRAM 341

G

glossary of terms and abbreviations 359

GLTGRP parameter
ADDCICSSIT 241
CHGCICSSIT 250
GLTLIB parameter
ADDCICSSIT 241
group list table 48
adding an entry 194
displaying an entry 196
removing an entry 198
working with entries 200

GROUP parameter
ADDCICSCVT 144
ADDCICSDCT 162
ADDCICSFCT 179
ADDCICSGLT 194
ADDCICSJCT 203
ADDCICSPCT 215
ADDCICSPPT 228
ADDCICSSIT 241
ADDCICSTCS 262
ADDCICSTCT 277
ADDCICSTST 296
CHGCICSCVT 150
CHGCICSDCT 169
CHGCICSFCT 185
CHGCICSGRP 137
CHGCICSJCT 207
CHGCICSPCT 219
CHGCICSPPT 232
CHGCICSSIT 250
CHGCICSTCS 266
CHGCICSTCT 285
CHGCICSTST 298
CRICICSGRP 136
DLTCICSGRP 138
DSPICSCVT 156
DSPICSDCT 174
DSPICSFCT 190
DSPICSGLT 196
DSPICSJCT 210
DSPICSPCT 222
DSPICSPPT 235
DSPICSSIT 258
DSPICSTCS 269
DSPICSTCT 291
DSPICSTST 300
INSCICSGRP 139
RMVICSCVT 157
RMVICSDCT 175
RMVICSFCT 191
RMVICSGLT 198
RMVICSJCT 211
RMVICSPCT 224
RMVICSPPT 237
RMVICSSIT 259
RMVICSTCS 271
RMVICSTCT 293
RMVICSTST 301
SAVCICSGRP 140
WRKCICSCVT 159

GROUP parameter (continued)
WRKCICSDCT 177
WRKCICSFCT 192
WRKCICSGLT 200
WRKCICSGRP 142
WRKCICSJCT 212
WRKCICSPCT 225
WRKCICSPPT 238
WRKCICSSIT 260
WRKCICSTCS 272
WRKCICSTCT 294
WRKCICSTST 303
group profile security 78
groups 46
changing description 137
creating 135
deleting 138
installing 138
saving 140
working with 141
GRTOBJAUT command 61, 78, 82

I

IDLETIME parameter
ADDCICSPCT 216
CHGCICSPCT 220
IMMEDIATE option
CEMT PERFORM SHUTDOWN 349
INDIRECT option
CEMT INQUIRE|SET
TDQUEUE 345
INDSYS parameter
ADDCICSTCS 264
CHGCICSTCS 268
initial menu security 76
initial program security 76
INQUIRE commands
AUTINSTMODEL 333
AUXTRACE 334
CONNECTION 335
FILE 336
INTRTRACE 338
JOURNALNUM 339
NETNAME 340
PROGRAM 312, 340
SYSTEM 342
TASK 343
TDQUEUE 344
TERMINAL 346
TRANSACTION 348
inquiring about resources 325
INSCICSGRP command 48, 138
INSERVICE option
CEMT INQUIRE|SET
CONNECTION 335
CEMT INQUIRE|SET
TERMINAL 346
INSGRP parameter
ADDCICSGLT 194
DSPICSGLT 197
RMVICSGLT 199
WRKCICSGLT 201
INSLIB parameter
ADDCICSGLT 194
DSPICSGLT 196
RMVICSGLT 198

INSLIB parameter (continued)
WRKCICSGLT 200
install group function, CEDA 49
install group, CEDA 58
installation 21
installation verification procedure 23
adding library to library list 24
copying the library 24
creating the CL program 28
journaling 28
map translation 25
resource definition 28
starting sample transactions 33
starting the control region 33
supplied resource definitions 28
translating programs 26
installing a group 48, 138
internal trace 104
interval control
additional IC shell 95
autoinstall terminals 96
batch shells 95
interval control elements 95
protected IC start requests 94
request identifiers 96
start requests 95
storage, lack of 95
task states 94
terminals, acquiring 95
timer-related tasks 94
interval control facility 94
INTRA option
CEMT INQUIRE|SET
TDQUEUE 345
intrapartition destination 85
intrapartition storage queues 43
INTRTRACE format
CEMT transaction 338
INTTRCTL parameter
ADDCICSSIT 243
CHGCICSSIT 252
INZCICS command 22
ITVCTL parameter
ADDCICSSIT 247
CHGCICSSIT 256

J

JFILE parameter
ADDCICSJCT 203
CHGCICSJCT 207
DSPICCSJCT 210
RMVICCSJCT 211
WRKCICSJCT 212
job description 124
job log 311, 312
JOB parameter 128
journal
record layout 96
records (user) 96
journal control
output synchronization 97
journal control table
adding an entry 202
changing an entry 206
displaying an entry 209
removing an entry 210

journal control table (*continued*)
 working with entries 212

journal files
 defining 96
 nonswitchable 97
 physical files 97
 switchable 97

journals, CEMT requests 339

journals, user
 creating 97
 local data area 98
 non-switchable 97
 prefix area 98
 restrictions 97
 switchable 97

JRNCTL parameter
 ADDCICSFCT 182
 CHGCICSFCT 188

JRNFILE parameter
 ADDCICSJCT 204
 CHGCICSJCT 208

JRNLIB parameter
 ADDCICSJCT 203
 CHGCICSJCT 207

JRNMBR parameter
 ADDCICSJCT 204
 CHGCICSJCT 209

JRNSTS parameter
 ADDCICSJCT 203
 CHGCICSJCT 207

JRN SWT parameter
 ADDCICSJCT 203
 CHGCICSJCT 207

K

KATAKANA parameter
 ADDCICSTCT 280
 CHGCICSTCT 288

key conversion 142, 146, 152

KEYINF parameter
 ADDCICSCVT 146
 CHGCICSCVT 152

keys, program function, PF 330

keys, tab 330

L

LCLQUEUE parameter
 ADDCICSPCT 217
 CHGCICSPCT 221

LENGTH option
 CEMT INQUIRE|SET
 PROGRAM 342

LENGTH parameter
 ADDCICSDCT 166
 CHGCICSDCT 172

LIB parameter
 ADDCICSCVT 144
 ADDCICSDCT 162
 ADDCICSFCT 179
 ADDCICSGLT 194
 ADDCICSJCT 202
 ADDCICSPCT 215
 ADDCICSPPT 227
 ADDCICSSIT 240

LIB parameter (*continued*)

ADDICSTCS 262
 ADDICSTCT 277
 ADDICSTST 296
 CHGCICSCVT 150
 CHGCICSDCT 168
 CHGCICSFCT 184
 CHGCICSGRP 137
 CHGCICSJCT 207
 CHGCICSPCT 219
 CHGCICSPPT 232
 CHGCICSSIT 250
 CHGCICSTCS 266
 CHGCICSTCT 284
 CHGCICSTST 298
 CRTICSGRP 136
 DLTCICSGRP 138
 DSPCICSCVT 155
 DSPCICSDCT 174
 DSPCICSFCT 190
 DSPCICSGLT 196
 DSPCICSJCT 210
 DSPCICSPCT 222
 DSPCICSPPT 235
 DSPCICSSIT 258
 DSPICSTCS 269
 DSPICSTCT 291
 DSPICSTST 300
 INSCICSGRP 139
 INZCICS 22

RMVICSCVT 157
 RMVICSDCT 175
 RMVICSFCT 191
 RMVICSGLT 198
 RMVICSJCT 211
 RMVICSPCT 224
 RMVICSPPT 236
 RMVICSSIT 259
 RMVICSTCS 270
 RMVICSTCT 292
 RMVICSTST 301
 SAVCICSGRP 140
 WRKCICSCVT 159
 WRKCICSDCT 176
 WRKCICSFCT 192
 WRKCICSGLT 200
 WRKCICSGRP 142
 WRKCICSJCT 212
 WRKCICSPCT 225
 WRKCICSPPT 238
 WRKCICSSIT 260
 WRKCICSTCS 272
 WRKCICSTCT 294
 WRKCICSTST 302

license manager 19

LIGHTPEN parameter
 ADDCICSTCT 281
 CHGCICSTCT 289

limited capability checking 76

locally-attached terminals 100

logical unit of work (LUW) 105

LOGOFF, CESF 124

M

main storage
 nonrecoverable TS queues 84

main storage (*continued*)
 recoverable TS queues 84
 temporary data 84

managing destination definitions 160

managing file definitions 177

managing group lists 193

managing groups 135

managing journal definitions 201

managing program definitions 226

managing system initialization
 definitions 239

managing system resource
 definitions 260

managing temporary storage
 definitions 295

managing terminal resource
 definitions 273

managing transaction definitions 213

MAPSET option
 CEMT INQUIRE|SET
 PROGRAM 342

master terminal operator (MTO)
 transaction, CEMT 325

MBR parameter
 ADDCICSDCT 165
 ADDCICSFCT 180
 CHGCICSDCT 171
 CHGCICSFCT 186

menu of CL commands 7

messages 9

migrating to a new release 22

MODE parameter
 ADDCICSTCS 262
 CHGCICSTCS 266

N

NAMEIND option
 CEMT INQUIRE|SET
 TDQUEUE 345

NETNAME format
 CEMT transaction 340

NETNAME option
 CEMT INQUIRE|SET
 CONNECTION 336
 CEMT INQUIRE|SET
 TERMINAL 347

NETWORK parameter
 ADDCICSTCS 262
 CHGCICSTCS 266
 CHGCICSTCT 286

networks
 CEMT transaction 340

NEWCOPY option
 CEMT INQUIRE|SET
 PROGRAM 342

NEWJRN parameter
 ADDCICSJCT 203
 CHGCICSJCT 208

NOADDABLE option
 CEMT INQUIRE|SET FILE 337

NOATI option
 CEMT INQUIRE|SET
 TERMINAL 347

NOBROWSE option
 CEMT INQUIRE|SET FILE 337

NOCEDF option
 CEMT INQUIRE|SET
 PROGRAM 342

NODELETE option
 CEMT INQUIRE|SET FILE 338

NOEMPTYREQ option
 CEMT INQUIRE|SET FILE 338

nonshared storage 67

NONSHRSTG parameter
 ADDCICSSIT 246
 CHGCICSSIT 255

NOREAD option 338

NOSWITCH option
 CEMT INQUIRE|SET
 AUXTRACE 334

NOTPURGEABLE option
 CEMT TRANSACTION 348

NOTTI option
 CEMT INQUIRE|SET
 TERMINAL 347

NOUPDATE option
 CEMT INQUIRE|SET FILE 338

number option
 CEMT INQUIRE|SET
 JOURNALNUM 339

O

object authority 76

OBJECT option
 CEMT INQUIRE|SET
 PROGRAM 341

object-oriented system 6

OBTAINING option
 CEMT INQUIRE|SET
 CONNECTION 336

online help 10

OPEN option
 CEMT INQUIRE|SET FILE 338
 CEMT INQUIRE|SET
 TDQUEUE 345

OPENOPTION parameter
 ADDCICSDCT 165
 CHGCICSDCT 171

OPTION parameter 119, 129

OS/400 command security 76

OS/400 journal 104

OS/400 license manager 19

OUTPUT option
 CEMT INQUIRE|SET
 JOURNALNUM 339

OUTPUT parameter
 DSPCICSCVT 157
 DSPCICSDCT 174
 DSPCICSFCT 190
 DSPCICSGLT 197
 DSPCICSJCT 210
 DSPCICSPCT 223
 DSPCICSPPT 236
 DSPCICSSIT 258
 DSPCICSTCS 270
 DSPCICSTCT 292
 DSPCICSTST 301

OUTSERVICE option
 CEMT INQUIRE|SET
 CONNECTION 336

OUTSERVICE option (*continued*)
 CEMT INQUIRE|SET
 TERMINAL 347

overtyping CEMT INQUIRE display 331

OVRPRTF CL command 103

P

passthru session 124

PERFORM commands
 SHUTDOWN 349
 SNAP 349

PGMID parameter
 ADDCICSPCT 216
 ADDCICSPPT 228
 CHGCICSPCT 220
 CHGCICSPPT 232
 DSPCICSPPT 235
 RMVCICSPPT 237
 WRKCICSPPT 238

PGMLNG parameter
 ADDCICSPPT 230, 234

PGMOBJ parameter
 ADDCICSPPT 228
 CHGCICSPPT 233

PGMSTS parameter
 ADDCICSPPT 229
 CHGCICSPPT 234

PHASEIN option
 CEMT INQUIRE|SET
 PROGRAM 342

PHYDEST parameter
 ADDCICSDCT 166
 CHGCICSDCT 172

plus sign in scrolling under CEMT 328

print spooling
 CRTPRTF command 102
 SPOOLOPEN command 102

printer spooling
 OVRPRTF CL command 103
 releasing spool files 103
 requirements 103

priority, control region 67

processing program table
 adding an entry 227
 changing an entry 231
 displaying an entry 235
 removing an entry 236
 working with entries 238

PROGRAM command
 CEMT transaction 340

program control table
 adding an entry 214
 changing an entry 218
 displaying an entry 222
 removing an entry 223
 working with entries 225

program function (PF) keys 330

PROGRAM option
 CEMT DISCARD 333
 CEMT INQUIRE|SET
 PROGRAM 342
 CEMT TRANSACTION 348

prompt screens 52

PRTICSTRC command 76

PRTFILE parameter
 ADDCICSTCT 278

PRTFILE parameter (*continued*)
 CHGCICSTCT 286

PURGE option
 CEMT INQUIRE|SET
 CONNECTION 336
 CEMT INQUIRE|SET TASK 343
 CEMT INQUIRE|SET
 TERMINAL 347

PURGE parameter
 ADDCICSPCT 216
 CHGCICSPCT 220

PURGEABLE option
 CEMT TRANSACTION 348

Q

question mark preceding CEMT 328

queues, CEMT requests 344

queues, temporary storage 84

queues, transient data 85

R

RCDACT parameter
 ADDCICSFCT 182
 CHGCICSFCT 188

RCDFMT parameter
 ADDCICSDCT 165
 ADDCICSFCT 180
 CHGCICSDCT 171
 CHGCICSFCT 186

RCVLMT parameter
 ADDCICSTCS 263
 CHGCICSTCS 268

RCVPFX parameter
 ADDCICSTCS 263
 CHGCICSTCS 267

RDO (resource definition online) 51, 58

READ option
 CEMT INQUIRE|SET FILE 338

RECLMT parameter
 ADDCICSJCT 203
 CHGCICSJCT 208

records, user journal 96

RECOVER parameter 32
 ADDCICSDCT 163
 ADDCICSFCT 181
 CHGCICSDCT 169
 CHGCICSFCT 187
 CRTICSGRP 136

recoverable files 90, 104
 IVP 28
 journaling 28

recoverable TS/TD files 86

recovery considerations 104

release-to-release compatibility 17, 21
 converting resource definitions 49, 50
 deleting groups 138
 SAVCICSGRP command 140
 WRKCICSxxx panel 54

RELEASED option
 CEMT INQUIRE|SET
 CONNECTION 336
 CEMT INQUIRE|SET
 TERMINAL 347

REMOTE option
 CEMT INQUIRE|SET FILE 338
 CEMT INQUIRE|SET
 TDQUEUE 345
 remote resources 42
 REMOTESYSTEM option
 CEMT INQUIRE|SET
 PROGRAM 342
 CEMT INQUIRE|SET
 TERMINAL 347
 removing resource definitions
 CVT 157
 DCT 175
 FCT 191
 GLT 198
 JCT 210
 PCT 223
 PPT 236
 SIT 259
 TCS 270
 TCT 292
 TST 301
 RESCOUNT option
 CEMT INQUIRE|SET
 PROGRAM 342
 resource checking 76
 resource definition
 ADDCICSSIT command 62
 DCT definition examples 86, 87
 device considerations 100
 dynamic installation 48, 49, 58, 139
 FCT definitions
 ESDS examples 92
 KSDS examples 91
 RRDS example 93
 file control considerations 88
 file types 42
 interval control considerations 94
 IVP 28
 JCT definition examples 100
 journal control considerations 96
 journal creation, example 99
 maintaining the group
 CHGCICSGRP 46
 CRTCICSGRP 46
 DLTCICSGRP 46
 SAVCICSGRP 46
 WRKCICSGRP 47
 PCT definition example 83
 physical files for TS/TD 63
 characteristics of file 63
 creating the TS/TD physical
 file 63
 CRTDUPOBJ command 64
 naming convention 63
 PPT definition example 83
 printer spooling considerations 102
 printer spooling example 103
 priority of control region 67
 program and map considerations 82
 prompt panel 53
 recommendations 63
 recoverability considerations 104
 remote resources 42, 102
 remote systems 101
 remote terminals 102
 SIT example 61

resource definition (*continued*)
 storage considerations 67
 supplied transaction examples 66
 supplied transactions, defining 65
 TCS definition example 101
 TCT definition examples 100
 temporary storage considerations 84
 trace considerations 104
 transaction considerations 82
 transient data considerations 85
 TST definition example 85
 user journals 97
 wait time, default 67
 work with group panel 55
 work with tables 56
 resource definition groups 46
 resource definition online (RDO) 51, 58
 resource definition online transaction,
 CEDA 51, 58
 resource definitions
 supplied 28
 resource security 75, 77
 resources, inquiring about 325
 RFILE parameter
 ADDCICSFCT 181
 RMTDEST parameter
 ADDCICSDCT 165
 CHGCICSDCT 172
 RMTDEV parameter
 ADDCICSTCT 278
 CHGCICSTCT 285
 RMTFILE parameter
 ADDCICSFCT 179
 CHGCICSFCT 185
 RMTKEYLEN parameter
 ADDCICSFCT 180
 CHGCICSFCT 185
 RMTLENGTH parameter
 ADDCICSFCT 180
 CHGCICSFCT 186
 RMTNETID parameter
 ADDCICSTCS 264
 CHGCICSTCS 268
 RMTPGMID parameter
 ADDCICSPPT 228
 CHGCICSPPT 232
 RMTQUEUE parameter
 ADDCICSTST 297
 CHGCICSTST 299
 RMTTRANSID parameter
 ADDCICSPCT 215
 CHGCICSPCT 219
 RMVCICSCVT command 157
 RMVCICSDCT command 175
 RMVCICSFCT command 191
 RMVCICSGLT command 198
 RMVCICSJCT command 210
 RMVCICSPCT command 223
 RMVCICSPPT command 236
 RMVCICSSIT command 259
 RMVCICSTCS command 270
 RMVCICSTCT command 292
 RMVCICSTST command 301
 routing transactions, CRTE 351
 RSRCID parameter
 ADDCICSCVT 144
 CHGCICSCVT 150

RSRCID parameter (*continued*)
 DSPCICSCVT 156
 RMVCICSCVT 158
 WRKCICSCVT 160
 running a batch job 33

S

sample program
 for autoinstall 309
 for IVP 23
 installing sample code 23
 to define a control region 34
 SAVCICSGRP command 22, 46, 140
 SAVF parameter
 SAVCICSGRP 141
 saving groups 140
 SBMJOB command 33
 SBMUSRJOB parameter
 ADDCICSJCT 204
 CHGCICSJCT 208
 SCRNSZE parameter
 ADDCICSPCT 217
 CHGCICSPCT 221
 scrolling a display 328
 security 6
 adoptive authority 78
 application programming
 commands 75
 autoinstall control program 311
 CICS application example 77
 communication 78
 control region setup 61, 82
 display station security 75
 EDTOBJAUT command 78
 group profile 78
 GRTOBJAUT command 78
 initial menu 76
 initial program 76
 limited capability 76
 object authority 76
 OS/400 command 76
 program object security 311
 resource 77
 resource checking 76
 resource security 75
 signon security 75
 supplied transactions 77
 system level 75
 system programming commands 75
 transaction 77
 user profile 78
 user transactions 77
 security considerations 75
 component information 81
 selection criteria 143, 147, 153
 server support for CICS clients 5
 SET commands
 AUTINSTMODEL 333
 AUXTRACE 334
 CONNECTION 335
 FILE 336
 INTRACE 338
 JOURNALNUM 339
 NETNAME 340
 PROGRAM 312, 340
 SYSTEM 342

- SET commands *(continued)*
 - TASK 343
 - TDQUEUE 344
 - TERMINAL 346
 - TRANSACTION 348
- SET FILE CLOSED
 - CEMT 89
 - CEMT command 337
 - system programming command 89
- shared open data paths 90
- shared storage 67
- shell
 - AEGSAMPLSO object 73
 - AEGSAMPYS object 73
 - AEGSAMPUSR object 74
 - internal tables 73
 - storage factors 72
- SHIP parameter
 - ADDCICSTCT 281
 - CHGCICSTCT 290
- SHRSTG parameter
 - ADDCICSSIT 245
 - CHGCICSSIT 254
- SHUTDOWN command
 - CEMT transaction 349
- shutting down a terminal shell 124
- signon security 75
- SITGRP parameter 113
- SITLIB parameter 113
- SLTCTL parameter
 - ADDCICSCVT 147
 - CHGCICSCVT 153
- SNAP command
 - CEMT transaction 349
- SNDLMT parameter
 - ADDCICSTCS 263
 - CHGCICSTCS 267
- SNDPEX parameter
 - ADDCICSTCS 263
 - CHGCICSTCS 267
- SOSI parameter
 - ADDCICSTCT 280
 - CHGCICSTCT 288
- space objects 67
- start journaling command 28
- START option
 - CEMT INQUIRE|SET
 - AUXTRACE 334
 - CEMT INQUIRE|SET
 - INTRTRACE 338
- STARTCODE option
 - CEMT INQUIRE|SET TASK 343
- STGDEV parameter
 - ADDCICSDCT 165
 - CHGCICSDCT 172
- STOP option
 - CEMT INQUIRE|SET
 - AUXTRACE 334
 - CEMT INQUIRE|SET
 - INTRTRACE 338
- storage considerations
 - control region 69
 - shell 72
- storage management 14
- storage queues for DCT 43
- storage requirements
 - CICS resource table sizes 69

- storage requirements *(continued)*
 - common work area 71
 - control region 68
 - size estimate for control region
 - AEGSAMPLSO object 69
 - AEGSAMPYS object 71
 - AEGSAMPUSR object 71
 - size estimate for shell
 - AEGSAMPLSO object 73
 - AEGSAMPYS object 73
 - AEGSAMPUSR object 74
 - supplied transactions 70, 74
- STRCICS 48
- STRCICS command 76, 112
 - CONVERT parameter 49, 50
- STRCICSUSR command 33, 76, 121, 125
- STRJRNP command 28
- STRPASTHR command 124
- STRTYPE parameter 113, 114
- supplied transactions
 - CEDA 51, 58
 - CEMT 325
 - CESF 129
 - CRTE 351
 - defining 65
 - online help for 323
 - security considerations 77
 - shell storage requirements 74
 - storage requirement 70
 - table of 65
- SWITCH option
 - CEMT INQUIRE|SET
 - AUXTRACE 334
- symbols used for syntax 323
- sync-level 2 6, 14
- syntax notation 323
- SYSID operand
 - CRTE transaction 351
- SYSID parameter
 - ADDCICSDCT 166
 - ADDCICSFCT 179
 - ADDCICSPCT 215
 - ADDCICSPPT 228
 - ADDCICSTCS 262
 - ADDCICSTCT 277
 - ADDCICSTST 297
 - CHGCICSDCT 172
 - CHGCICSFCT 185
 - CHGCICSPCT 219
 - CHGCICSPPT 232
 - CHGCICSTCS 266
 - CHGCICSTCT 285
 - CHGCICSTST 299
 - DSPICSTCS 269
 - RMVICSTCS 271
 - WRKICSTCS 272
- SYSID value 113, 126
- SYSSTS parameter
 - ADDCICSTCS 263
 - CHGCICSTCS 267
- system administrator
 - duties 324
- system initialization table
 - adding an entry 239
 - changing an entry 248
 - displaying an entry 258
 - removing an entry 259

- system initialization table *(continued)*
 - working with entries 260
- system storage space 67
- system-level security 75

T

- tab keys 330
- table types 42
- task
 - automatic file closure 90
- TASK option
 - CEMT INQUIRE|SET TASK 344
- tasks, CEMT requests 343
- TCPPORT parameter
 - ADDCICSSIT 248
 - CHGCICSSIT 257
- TDCTL parameter
 - ADDCICSSIT 247
 - CHGCICSSIT 256
- TDQUEUE option
 - CEMT INQUIRE|SET
 - TDQUEUE 345
- templates, data conversion 142
- temporary storage 11
 - auxiliary 84
 - main 84
 - queue 84
- temporary storage data 84
- temporary storage table
 - adding an entry 296
 - changing an entry 298
 - displaying an entry 300
 - removing an entry 301
 - working with entries 302
- TERM option
 - CEMT INQUIRE|SET TASK 344
- TERMIN option
 - CEMT INQUIRE|SET
 - TDQUEUE 345
- terminal control
 - autoinstall terminals, defining 308
 - autoinstall terminals, generating
 - terminal identifiers 307
 - remote considerations 102
 - TCT entries 100
- terminal control system entry table
 - adding an entry 261
 - changing an entry 265
 - displaying an entry 269
 - removing an entry 270
 - working with TCS entries 272
- terminal control table
 - adding an entry 275
 - changing an entry 282
 - displaying an entry 291
 - removing an entry 292
 - working with entries 294
- terminals
 - CEMT INQUIRE|SET requests 346
- terminals supported 17
- terminals, automatic installation 312
- TEXT parameter
 - CHGCICSGRP 137
 - CRTICSGRP 136
- TGTRLS parameter
 - SAVICSGRP 141

TITLE option
 CEMT PERFORM SNAP 349

trace
 printing 104

TRANID option
 CEMT INQUIRE|SET TASK 344
 CEMT INQUIRE|SET
 TDQUEUE 345
 CEMT TRANSACTION 348

TRANID parameter 126

transaction identifiers, entering 54

TRANSACTION option
 CEMT DISCARD 333
 CEMT INQUIRE|SET
 TERMINAL 347

transaction routing, CRTE 351

TRANSID parameter
 ADDCICSDCT 164
 ADDCICSPCT 215
 ADDCICSPPT 229
 ADDCICSTCT 280
 CHGCICSDCT 170
 CHGCICSPCT 219
 CHGCICSPPT 234
 CHGCICSTCT 288
 DSPCICSPCT 223
 RMVCICSPCT 224
 WRKCICSPCT 225

transient data 11
 intrapartition destinations 85

transient data control
 extrapartition destinations 85

translating BMS maps 25

translating the autoinstall control
 program 310

TRANSTS parameter
 ADDCICSPCT 216
 CHGCICSPCT 220

TRGLVL parameter
 ADDCICSDCT 163
 CHGCICSDCT 170

TRIGGERLEVEL option
 CEMT INQUIRE|SET
 TDQUEUE 345

TSCTL parameter
 ADDCICSSIT 244
 CHGCICSSIT 254

TSQUEUE parameter
 ADDCICSTST 296
 CHGCICSTST 298
 DSPCICSTST 300
 RMVCICSTST 302
 WRKCICSTST 303

TTI option
 CEMT INQUIRE|SET
 TERMINAL 347

TTISTS parameter
 ADDCICSTCT 279
 CHGCICSTCT 287

TWASIZE parameter
 ADDCICSPCT 217
 CHGCICSPCT 221

two-phase commit 6

TYPE parameter
 ADDCICSDCT 163
 ADDCICSTST 296
 CHGCICSDCT 169

TYPE parameter (*continued*)
 CHGCICSTST 299

U

UCTRN parameter
 ADDCICSPCT 217
 ADDCICSTCT 281
 CHGCICSPCT 221
 CHGCICSTCT 289

UNATTEND parameter
 ADDCICSTCT 280
 CHGCICSTCT 289

UPDATE option
 CEMT INQUIRE|SET FILE 338

USECOUNT option
 CEMT INQUIRE|SET
 PROGRAM 342

user journaling 96, 104

user profile security 78

user shell
 CESF LOGOFF 124
 controlled shutdown 124
 data queue input 123
 initialization and termination
 table 122, 124
 link to application program 123
 shell environment 121
 transaction scheduling 121
 types of shells 121
 work flow
 initialization 122
 runtime processing 123
 shutdown 124

user storage space 67

user transactions
 security considerations 77

user-based pricing 19
 counting users 19
 definition 19
 tasks not counted as users 20
 user, definition of 19

USERID option
 CEMT INQUIRE|SET TASK 344

USRARASIZE parameter
 ADDCICSTCT 279
 CHGCICSTCT 287

USRJOB parameter
 ADDCICJCT 204
 CHGCICJCT 208

USRTRC parameter
 ADDCICSSIT 244
 CHGCICSSIT 253

V

VALIDATION parameter
 ADDCICSTCT 281
 CHGCICSTCT 289

VSAM
 alternate index 89

VSAM emulation 88

VSAM option
 CEMT INQUIRE|SET FILE 338

VTAM terminals, acquiring 346

W

wait time, default 67

WAITTIME parameter
 ADDCICSPCT 216
 CHGCICSPCT 220

warm start 113, 115

work management 6

working with an active job 33

working with resource definitions
 CVT 159
 DCT 176
 FCT 192
 GLT 200
 groups 141
 JCT 212
 PCT 225
 PPT 238
 SIT 260
 TCS 272
 TCT 294
 TST 302

WRKACTJOB command 33

WRKARASIZE parameter
 ADDCICSSIT 241
 CHGCICSSIT 250

WRKCICSCVT command 159

WRKCICSDCT command 176

WRKCICSFCT command 192

WRKCICSGLT command 200

WRKCICSGRP command 47, 141

WRKCICJCT command 212

WRKCICSPCT command 225

WRKCICSPPT command 238

WRKCICSSIT command 260

WRKCICSTCS command 272

WRKCICSTCT command 294

WRKCICSTST command 302



Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC41-5455-00

