CICS Transaction Server for z/OS
5.5

CICSPlex SM API Reference

Note

Before using this information and the product it supports, read the information in "Notices" on page 175.

This edition applies to the IBM® CICS® Transaction Server for z/OS® Version 5 Release 5 (product number 5655-Y04) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this PDF

This PDF is a reference of the commands of the application programming interface for the CICSPlex SM element of CICS Transaction Server for z/OS. This documentation is intended for application programmers who are writing applications to interact with CICSPlex SM.

For information about how to write applications for CICS, using this API, see *CICSPlex SM Application Programming Guide*.

For details of the terms and notation used in this book, see Conventions and terminology used in the CICS documentation in IBM Knowledge Center.

Date of this PDF

This PDF was created on 2024-04-22 (Year-Month-Date).

Chapter 1. Using the CICSPlex SM command-level interface

This section describes how to use the CICSPlex® SM command-level interface.

CICSPlex SM command format

The format of an API command when issued through the CICSPlex SM command-level interface is EXECUTE CPSM (or EXEC CPSM) followed by the name of the required command and possibly by one or more options.

The syntax is as follows:

```
EXEC CPSM command option(arg)....
```

where:

command
Describes the operation required (for example, CONNECT).

option
Describes any of the required or optional facilities available with each command. Some options are followed by an argument in parentheses. You can write options (including those that require arguments) in any order.

arg
Which is short for argument, is a value such as *data-value* or *data-ref*. A *data-value* can be a constant. This means that an argument that sends data to CICSPlex SM is generally a *data-value*. An argument that receives data from CICSPlex SM must be a *data-ref*.

Here is an example of an EXEC CPSM command:

```
EXEC CPSM CONNECT
          USER(JONES) VERSION(0310)
          CONTEXT(EYUPLX01) SCOPE(EYUCSG01)
          THREAD(THRDTKN)
          RESPONSE(RESPVAR) REASON(REASVAR)
```

You must add an end-of-command delimiter that is valid for the programming language you are using. In COBOL programs, for example, the end-of-command delimiter is an END-EXEC statement. In PL/I and C programs, the delimiter is a semicolon (;).

CICSPlex SM argument values

You must specify the parenthesized argument values that follow options in an API command.

These values are as follows:

data-value
A sending argument that is used to pass data from your program to CICSPlex SM.

The data you pass can be fullword binary data, fixed or variable length character data, or unspecified. If the data type is unspecified, CICSPlex SM assumes a composite data structure made up of multiple fields of varying data types. The argument can be in one of these forms:

- Variable name
- Self-defining term
- Expression

data-value includes *data-ref* as a subset.

data-ref
A receiving (or sending and receiving) argument that is used primarily to pass data from CICSPlex SM to your program.

The data type can be any of the same types allowed for *data-value* arguments. However, the argument must be a named variable.

In some cases you can use a *data-ref* argument to provide input to CICSPlex SM before CICSPlex SM returns its output to you; for example, you could specify a *data-ref* argument on the COUNT option of the FETCH command.

data-area
A sending or receiving argument that is used to identify a buffer that contains data. A *data-area* argument can be considered to be a *data-ref* argument with an unspecified data type. A *data-area* argument cannot be defined by a self-defining term or expression; it must be a named variable.

ptr-ref
A receiving argument that is used to pass pointer values from CICSPlex SM to your program.

A *ptr-ref* argument is a special form of *data-ref* argument. The data that is being passed is an address pointer, rather than binary or character data.

cpsm-token
A sending or receiving argument that is used to pass identifying tokens that are generated by CICSPlex SM. A *cpsm-token* argument can be considered to be a *data-ref* argument with an unspecified data type.

Tokens are created by CICSPlex SM to identify API processing threads, result sets, filters, and notifications.

Because token values are created by CICSPlex SM, your program must receive a token into a variable before it can specify that token on subsequent commands. A token cannot be defined by a self-defining term or expression; it must be a named variable.

COBOL argument values

In COBOL, you can replace the argument values of the CICSPlex SM API with different data types. The data type that you can use depends on the type of argument value.

data-value
Can be replaced by any COBOL data name of the correct data type for the argument, or by a constant that can be converted to the correct type for the argument. The table that follows indicates how to define the correct data type:

Data type	**COBOL definition**
Halfword binary	PIC S9(4) USAGE BINARY
Fullword binary	PIC S9(8) USAGE BINARY
Doubleword binary	PIC S9(18) COMP
Pointer	USAGE IS POINTER
Character string	PIC X(*n*) where *n* is the number of bytes
UTF-8 character string	PIC X(*n*) where *n* is the number of bytes
Packed decimal	PIC S9(*n*) COMP-3 where *n* is the number of decimal digits

data-value includes *data-ref* as a subset.

data-ref
Can be replaced by any COBOL data name of the correct data type for the argument. The table that follows indicates how to define the correct data type:

Data type	COBOL definition
Halfword binary	PIC S9(4) USAGE BINARY
Fullword binary	PIC S9(8) USAGE BINARY
Doubleword binary	PIC S9(18) COMP
Pointer	USAGE IS POINTER
Character string	PIC X(*n*) where *n* is the number of bytes
UTF-8 character string	PIC X(*n*) where *n* is the number of bytes
Packed decimal	PIC S9(*n*) COMP-3 where *n* is the number of decimal digits

Where the data type is unspecified, *data-ref* can refer to an elementary or group item.

data-area
Can be replaced by any COBOL data name with a data type of halfword binary (PIC S9(4) COMP), fullword binary (PIC S9(8) COMP), or character string (PIC X(*n*)).

ptr-ref
Can be replaced by a pointer variable or an ADDRESS special register.

cpsm-token
Can be replaced by any COBOL data name with a data type of fullword binary (PIC S9(8) COMP).

C argument values

In C, you can replace the argument values of the CICSPlex SM API with different data types. You can use any data reference of the correct data type for a *data-value*, *data-ref*, or *data-area* provided the reference is to contiguous storage.

data-value
Can be replaced by any C expression that can be converted to the correct data type for the argument. The table that follows indicates how to define the correct data type:

Data type	C definition
Halfword binary	short int
Fullword binary	long int
Doubleword binary	long long int
Character array	char[*n*] where *n* is the number of bytes in the field (the field must be padded with blank spaces)
UTF-8 character string	char[*n*] where *n* is the number of bytes in the field (the field is padded with ASCII blanks)

data-value includes *data-ref* as a subset.

data-ref
Can be replaced by any C data reference that has the correct data type for the argument. The table that follows indicates how to define the correct data type:

Data type	C definition
Halfword binary	short int
Fullword binary	long int
Doubleword binary	long long int

Data type	C definition
Character array	char[*n*] where *n* is the number of bytes in the field (the field must be padded with blank spaces)
UTF-8 character string	char[*n*] where *n* is the number of bytes in the field (the field is padded with ASCII blanks)

If the data type is unspecified, *data-ref* can refer to a scalar data type, array, or structure. The reference must be to contiguous storage.

data-area
Can be replaced by any named variable with a data type of halfword binary (short int), fullword binary (long int), or character array (char[*n*]).

ptr-ref
Can be replaced by any C pointer type reference.

cpsm-token
Can be replaced by any named variable with a data type of fullword binary (long int).

PL/I argument values

In PL/I, you can replace the argument values of the CICSPlex SM API with any PL/I data reference of the correct data type, provided the reference is to connected storage.

data-value
Can be replaced by any PL/I expression that can be converted to the correct data type for the argument. The table that follows indicates how to define the correct data type:

Data type	PL/I definition
Halfword binary	FIXED BIN(15)
Fullword binary	FIXED BIN(31)
Doubleword binary	FIXED BIN(63)
Pointer	POINTER
Character string	CHAR(*n*) where *n* is the number of bytes
UTF-8 character string	CHAR(*n*) where *n* is the number of bytes
Packed decimal (*n* decimal digits)	FIXED DEC(*n*,0)

data-value includes *data-ref* as a subset.

data-ref
Can be replaced by any PL/I data reference that has the correct data type for the argument. The table that follows indicates how to define the correct data type:

Data type	PL/I definition
Halfword binary	FIXED BIN(15)
Fullword binary	FIXED BIN(31)
Doubleword binary	FIXED BIN(63)
Pointer	POINTER
Character string	CHAR(*n*) where *n* is the number of bytes
UTF-8 character string	CHAR(*n*) where *n* is the number of bytes

Data type	PL/I definition
Packed decimal (*n* decimal digits)	FIXED DEC(*n*,0)

If the data type is unspecified, *data-ref* can refer to an element, array, or structure; for example, FROM(P–>STRUCTURE) LENGTH(LNG). The reference must be to connected storage.

The data area must also have the correct PL/I alignment attribute: ALIGNED for binary items, and UNALIGNED for strings.

If you use a varying data string without an explicit length, the data passed begins with two-byte length fields, and its length is the maximum length declared for the string. If you explicitly specify a length in the command, the data passed has this length; that is, the two-byte length fields followed by data up to the length you specified.

data-area
Can be replaced by any named variable with a data type of halfword binary (FIXED BIN(15)), fullword binary (FIXED BIN(31)), or character string (CHAR(*n*)).

ptr-ref
Can be replaced by any PL/I reference of type POINTER ALIGNED.

cpsm-token
Can be replaced by any named variable with a data type of fullword binary (FIXED BIN(31)).

Assembler language argument values

In general, an argument can be either the address of the data or the data itself (in assembler language terms, either a relocatable expression or an absolute expression).

A relocatable expression must not contain unmatched brackets (outside quotation marks) or unmatched quotation marks (apart from length-attribute references). If this rule is obeyed, any expression can be used, including literal constants, such as =AL2(100), forms such as 20(0,R11), and forms that use the macro-replacement facilities.

An absolute expression must be a single term that is either a length-attribute reference, or a self-defining constant.

Care must be taken with equated symbols, which should be used only when referring to registers (pointer references). If an equated symbol is used for a length, for example, it is treated as the address of the length and an unpredictable error occurs.

The argument values can be replaced as follows:

data-value
Can be replaced by a relocatable expression that is an assembler language reference to data of the correct type for the argument, or by a constant of the correct type for the argument.

data-ref
Can be replaced by a relocatable expression that is an assembler language reference to data of the correct type for the argument.

data-area
Can be replaced by a relocatable expression that is an assembler language reference to data with a type of halfword (DS H), fullword (DS F), or character string (CLn).

ptr-ref
Can be replaced by any absolute expression that is an assembler language reference to a register.

cpsm-token
Can be replaced by a relocatable expression that is an assembler language reference to data with a type of fullword (DS F).

Using CICSPlex SM with REXX

You can invoke CICSPlex SM API commands from a REXX program.

Command format

An API command can be passed from REXX to CICSPlex SM either by using the REXX ADDRESS command or by using the EYUAPI() function.

The format of the REXX ADDRESS command is as follows:

```
ADDRESS CPSM 'command option(arg)...'
```

This method of calling the API invokes a CICSPlex SM host subcommand environment.

Alternatively, you can use the EYUAPI() function supplied by CICSPlex SM:

```
var = EYUAPI('command option(arg)...')
```

This method invokes the CICSPlex SM REXX function package.

Note that with both methods you can enter text in either upper or lower case.

Here is an example of an API command as it would be issued from a REXX program:

```
var = EYUAPI('CONNECT'              ,
             'CONTEXT('WCONTEXT')' ,
             'SCOPE('WSCOPE')'     ,
             'VERSION(0550)'      ,
             'THREAD(THRDTKN)'     ,
             'RESPONSE(RESPVAR)' ,
             'REASON(REASVAR)')
   .
   .
```

Argument values

The CICSPlex SM API makes full use of the standard REXX variable interface.

A *data-value* argument is considered to be character input. Binary data (including EYUDA and CVDA values) is translated into the appropriate internal format. User tokens are not translated.

data-ref
A receiving (or sending and receiving) argument used primarily to pass data from CICSPlex SM to your program.

A *data-ref* argument must be a named variable that can be used to receive the resulting output. The output data is translated as appropriate:

- Character data is not translated; the data is placed into the variable as is.
- Binary data is translated to display format (decimal) and placed into the variable.
- User tokens are not translated; the token value is placed into the variable as is.
- Address values are not translated; the specified storage buffer is placed directly into one or more variables.

In some cases, you can use a *data-ref* argument to provide input to CICSPlex SM before CICSPlex SM returns its output to you (the COUNT option on the FETCH command is an example of this). If a *data-ref* argument can be supplied as input, you must specify a variable for that argument. If you do not want to specify an input value, you should initialize the variable.

data-area
A sending or receiving argument used to identify a buffer that contains data. A *data-area* argument must be a named variable.

For output buffers that could receive multiple resource table records, CICSPlex SM creates (or fills) stem variables to hold the data. The zero entry of the stem array indicates the number of entries in the array.

For example, in the stem variable called W_INTO_EVALDEF, the W_INTO_EVALDEF.0 entry contains the number of EVALDEF resource table records returned. The entries W_INTO_EVALDEF.1 through W_INTO_EVALDEF.n contain the actual resource table records.

A stem variable is created regardless of whether the actual output is a single record or multiple records.

ptr-ref
A receiving argument used to pass pointer values from CICSPlex SM to your program.

A *ptr-ref* argument must be a named variable that can be used to receive the resulting output. The data being passed is a character representation of a hexadecimal address.

cpsm-token
A sending or receiving argument used to pass identifying tokens that are generated by CICSPlex SM.

A *cpsm-token* argument must be a named variable. Tokens are not translated; the token value is placed into the variable as is.

Note: Each variable (or stem variable) returned by CICSPlex SM contains an entire resource table record. You can use the TPARSE command to break a record into individual fields. For a description of this command, see Chapter 2, “REXX functions and commands,” on page 11.

MVS restrictions

A number of general restrictions apply to all CICSPlex SM API commands.

- The program must be in primary addressing mode when invoking any CICSPlex SM service. The primary address space must be the home address space. All parameters passed to CICSPlex SM must reside in the primary address space.
- CICSPlex SM does not always preserve access registers across commands. If your program uses access registers, it should save them before invoking a CICSPlex SM service, and restore them before reusing them.

CICS and CICSPlex SM value data areas

The values for some CICSPlex SM resource table attributes are maintained in an encoded form.

These values can be CICSPlex SM value data areas (EYUDAs) or CICS value data areas (CVDAs). You can use one of two built-in translator functions to translate these values:

EYUDAs
Use the CICSPlex SM translator function called EYUVALUE.

The EYUVALUE function is not available to programs written in REXX. You can use the **TPARSE** command, that is supplied specifically for REXX programs, to access and translate the attribute values in a resource table. For a description of this command, see Chapter 2, “REXX functions and commands,” on page 11.

CVDAs
Use the CICS translator function called DFHVALUE.

In some CICS environments, the DFHVALUE function might return incompatible CVDA values. Because these CVDA values conflict with values used in other CICS environments, CICSPlex SM must modify them to retain their uniqueness. CICSPlex SM adds either 8000 or 9000 to the value returned by DFHVALUE for each of these CVDA attributes. For more information about translating CVDA values, see TRANSLATE command.

For example, consider the following COBOL statement:

```
MOVE EYUVALUE(QUIESCING) TO EYUDATA
```

This statement translates the EYUDA character value of QUIESCING into its numeric equivalent of 48 when the program is translated. CICSPlex SM also provides a **TRANSLATE** command to translate EYUDA and CVDA values at run time. You can use **TRANSLATE** to convert an EYUDA or CVDA value that is associated with a specific resource table and attribute:

```
EXEC CPSM TRANSLATE OBJECT(WLMAWAOR)
                    ATTRIBUTE(STATUS)
                    FROMCV(48)
                    TOCHAR(EYUCHAR)
                    RESPONSE(RESPDATA)
                    REASON(REASDATA)
```

This command translates the EYUDA value for the STATUS attribute of the WLMAWAOR resource table into its character value when the program runs.

For a description of the **TRANSLATE** command, see TRANSLATE command. For a list of the EYUDA values used by CICSPlex SM, see Chapter 7, “EYUDA values,” on page 137.

Language considerations

All of the language considerations that apply to the various environments (CICS, MVS™ batch, TSO, and NetView®) also apply to CICSPlex SM programs written to run in those environments.

Length options

Many API commands involve the transfer of data between the application program and CICSPlex SM.

In COBOL, PL/I, and Assembler language, the translator can default certain length options; this means they may be optional in programs that specify data areas. In C and REXX, all length options must be specified.

The CICSPlex SM API allows most `data-value` arguments, which are only passed from your program to CICSPlex SM, to default. The exception is the LENGTH option on the following commands:

- CREATE
- REMOVE
- UPDATE

On the other hand, `data-ref` arguments, which can be passed from your program to CICSPlex SM and back again, must always be specified.

When an API command offers a length option, it is always expressed as a signed fullword binary value. This puts a theoretical upper limit of 2 147 483 647 bytes on the length. The achievable upper limit varies from command to command and with various language compilers, but the maximum limit of all input data areas on an API command is typically 16 124 bytes. When this limit is exceeded the API command fails with a response of INVALIDCMD and a reason of LENGTH.

RESPONSE and REASON options

Once an API command completes processing, it returns a response and, if appropriate, a reason. You must specify the RESPONSE and REASON options on each command to receive the response and reason values returned by that command.

Note: The TBUILD and TPARSE commands, which can be used only with the REXX run-time interface, do not use the RESPONSE and REASON options. The result of these REXX-specific processes is returned by their STATUS option. For more information, see the descriptions of the TBUILD and TPARSE commands in Chapter 2, “REXX functions and commands,” on page 11.

RESPONSE*(data-ref)*

data-ref is a user-defined variable. On return from the command, it contains a character value that describes the result of command processing. RESPONSE values are given in the description of each command.

REASON*(data-ref)*

data-ref is a user-defined variable. On return from the command, it contains a value that further qualifies the response to certain commands. REASON values are given with the RESPONSE values, for those responses that use them.

For more information about the RESPONSE and REASON options, see Developing CICSPlex SM applications. For a summary of RESPONSE and REASON values by command, see Chapter 6, "RESPONSE and REASON values," on page 129.

Chapter 2. REXX functions and commands

This section contains detailed descriptions of the REXX functions and commands supplied with CICSPlex SM. These functions and commands can be used only with the REXX run-time interface.

Each description includes the following

- A description of the command
- Purpose
- Syntax of command (*var* represents a variable)
- Available options for the command
- REXX response codes returned by the command

The functions are presented in alphabetical order:

Functions

The REXX functions supplied with CICSPlex SM make use of standard REXX variable substitution rules.

In addition to REXX return codes, these functions can produce EYUARnnnn messages. For descriptions of those messages, see CICSPlex SM messages.

EYUAPI()

Pass an API command to CICSPlex SM.

```
var = EYUAPI(command string)
    OR
var = EYUAPI('command string')
```

Description

This function passes an API command to CICSPlex SM. You must issue an EYUAPI or EYUINIT function before you can use the ADDRESS CPSM command to pass API commands to REXX.

Options

command string
Identifies the API command and options to be passed.

Return codes

The following is a list of the REXX return codes that can be returned by the EYUAPI function in its assigned variable (*var*).

These return codes indicate what REXX did with the EYUAPI function; they do not indicate whether the API command that was passed was successfully processed by CICSPlex SM. For that information, you must refer to the RESPONSE and REASON values returned by the command.

0
The EYUAPI function was successful.

1
The EYUAPI function failed.

EYUINIT()

Initialize the CICSPlex SM API environment and allocate the necessary resources.

```
var = EYUINIT()
```

Description

This command initializes the CICSPlex SM API environment and allocates the necessary resources. EYUINIT should be the first function issued in a REXX program.

Note: You must issue an EYUINIT or EYUAPI function before you can use the ADDRESS CPSM command to pass API commands to REXX.

Return codes

The following is a list of the REXX return codes that can be returned by the EYUINIT function in its assigned variable (*var*).

0
The EYUINIT function was successful.

1
The EYUINIT function failed.

EYUREAS()

Translate the numeric value returned by the REASON option of an API command.

```
var = EYUREAS(reason)
```

Description

This command translates the numeric value returned by the REASON option of an API command into its character equivalent and vice versa.

Options

reason
Is the REASON value to be translated.

Return codes

The following is a list of the REXX return codes that can be returned by the EYUREAS function in its assigned variable (*var*).

nnnn
The numeric or character equivalent of the specified REASON value.

-1
The specified REASON value is invalid and could not be translated.

EYURESP()

Translate the numeric value returned by the RESPONSE option of an API command.

```
var = EYURESP(response)
```

Description

This command translates the numeric value returned by the RESPONSE option of an API command into its character equivalent and vice versa.

Options

response
Is the RESPONSE value to be translated.

Return codes

The following is a list of the REXX return codes that can be returned by the EYURESP function in its assigned variable (*var*).

nnnn
The numeric or character equivalent of the specified RESPONSE value.

-1
The specified RESPONSE value is invalid and could not be translated.

EYUTERM()

Terminate the CICSPlex SM API environment and release any allocated resources.

```
var = EYUTERM()
```

Description

This command terminates the CICSPlex SM API environment and releases any allocated resources. EYUTERM should be the last function issued in a REXX program.

Note: If the CICSPlex SM host subcommand environment is installed at your enterprise (as opposed to being called from the function package), you may not need to use EYUTERM at the end of every program. The resources that remain allocated can be reused by the next program that accesses the host subcommand environment.

Return codes

The following is a list of the REXX return codes that can be returned by the EYUTERM function in its assigned variable (*var*).

0
The EYUTERM function was successful.

1
The EYUTERM function failed.

Commands

The REXX-specific commands supplied with CICSPlex SM perform a series of API commands internally and return the results to REXX.

The commands are presented here in alphabetical order. Each description includes the purpose, syntax, and available options for the command.

Note:

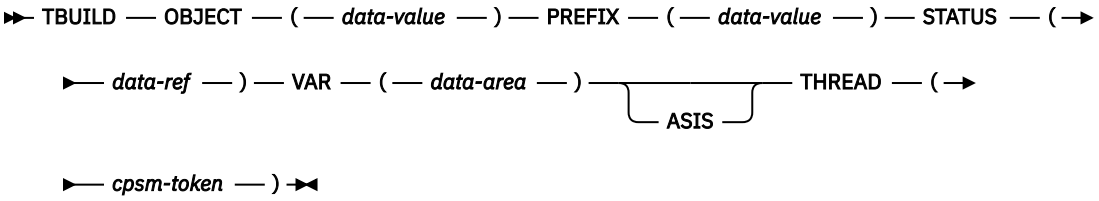
1. You cannot use these commands to process user-defined views of a resource table that were created by the SPECIFY VIEW command. If you create a view with the same name as a supplied resource table and then specify that name on one of these commands, the command fails.
2. These commands do not use the RESPONSE and REASON options. The result of these REXX-specific processes is returned by the STATUS option.
3. These commands do not provide any useful FEEDBACK information. The API commands that are issued internally reuse the same feedback area. So, when one of these commands finishes processing, the feedback area does not represent the entire sequence of events.

The commands are:

- "TBUILD" on page 14
- "TPARSE" on page 16

TBUILD

Build a resource table record from a set of variables.



Description

This command builds a resource table record from a set of variables that represent the individual attributes of a CICSPlex SM or CICS definition. A definition is represented by a resource table with a type of CPSM Definition or CICS Definition.

You form the attribute variables by adding a prefix to the attribute name, like this:

```
prefix_attribute
```

where `prefix` is a text string that you supply and `attribute` is the name of an attribute in the resource table. You must insert an underscore character (_) between the prefix and the attribute name.

The resource table record can be placed in any valid REXX variable, including a stem variable.

TBUILD only uses the attributes that you specify; it does not assume any default values for optional attributes. If you do not supply a variable for an attribute that is optional, the corresponding field in the

resource table record is initialized according to its data type (that is, character fields are set to blanks, binary data and EYUDA values are set to zeroes).

Note: For a list of the CICSPlex SM resource tables by type, see Developing CICSPlex SM applications. For a complete description of a particular resource table and its attributes, see the CICSPlex SM resource tables in Reference.

Options

ASIS
Indicates that the resource table attribute values are already in their internal format; they are to be processed as is, rather than translated.

You must use the ASIS option to rebuild a CICSPlex SM or CICS definition that you previously parsed (with the TPARSE ASIS command).

OBJECT*(data-value)*
Identifies the resource table for which a record is to be built. This value must be the 1- to 8-character name of a valid CPSM Definition or CICS Definition resource table.

Note: You cannot use the TBUILD command to process a resource table view that was created by the SPECIFY VIEW command. If you create a view with the same name as a supplied resource table and then specify that name on a TBUILD command, the command fails.

PREFIX*(data-value)*
Specifies the prefix you used to name the variables that contain the resource table attributes.

Note: The maximum allowable length for a prefix is determined by REXX and the environment in which the program runs.

STATUS*(data-ref)*
Names a variable to receive the REXX status value returned for this command. The status is returned in character form as one of the following:

OK
The TBUILD command completed processing successfully.

SYNTAX ERROR
The TBUILD command could not be processed because of a syntax error. EYUARnnnn messages that describe the error are written to the destination defined on your system for IRXSAY WRITEERR output.

FAILURE
The TBUILD command failed because some of the data it was attempting to process is invalid. Trace data is written to a REXX stem variable called EYUTRACE. EYUARnnnn messages that describe the failure may also be written to the destination defined on your system for IRXSAY WRITEERR output.

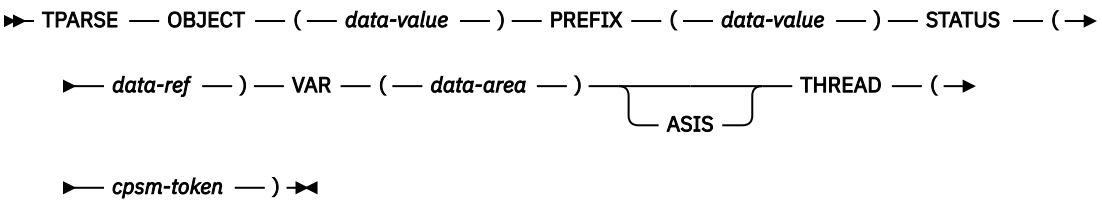
Note: For more information about the EYUTRACE stem variable, see Developing CICSPlex SM applications.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

VAR*(data-area)*
Names a variable to receive the resource table record that is built by TBUILD.

TPARSE

Parse a resource table record from a variable into a set of variables.



Description

This command parses a resource table record from a variable into a set of variables that represent the individual attributes of the table. You can use TPARSE with any type of CICSPlex SM resource table.

The resource table variable can be any valid REXX variable, including a stem variable. The output variables are formed by adding a prefix to the attribute name, like this:

```
prefix_attribute
```

where `prefix` is a text string that you supply and `attribute` is the name of an attribute in the resource table. An underscore (_) is inserted between the prefix and the attribute name.

Note: For complete descriptions of the resource tables and their attributes, see the CICSPlex SM resource tables.

Options

ASIS
Specifies that the resource table attribute values are not to be translated into their external format; they are to be returned as is. Attribute values are presented as follows:

- Character values have trailing blanks.
- Binary values have leading zeroes and are not converted to display format.
- EYUDA and CVDA values are not converted to character format.

You must use the ASIS option to parse a CPSM Definition or CICS Definition resource table that you want to rebuild (with the TBUILD ASIS command).

Note: If you use the ASIS option with EYUDA or CVDA values, you can use the TRANSLATE command to convert the coded numeric value into a character value.

OBJECT*(data-value)*
Identifies the resource table that is to be parsed. This value must be the 1- to 8-character name of a valid resource table.

Note: You cannot use the TPARSE command to process a resource table view that was created by the SPECIFY VIEW command. If you create a view with the same name as a supplied resource table and then specify that name on a PARSE command, the command fails.

PREFIX*(data-value)*
Specifies the prefix you want to use to name the attribute variables returned by TPARSE.

Note: The maximum allowable length for a prefix is determined by REXX and the environment in which the program runs.

STATUS*(data-ref)*
Names a variable to receive the REXX status value returned for this command. The status is returned in character form as one of the following:

OK
The TPARSE command completed processing successfully.

SYNTAX ERROR
The TPARSE command could not be processed because of a syntax error. EYUARnnnn messages that describe the error are written to the destination defined on your system for IRXSAY WRITEERR output.

FAILURE
The TPARSE command failed because some of the data it was attempting to process is invalid. Trace data is written to a REXX stem variable called EYU_TRACE. EYUARnnnn messages that describe the failure may also be written to the destination defined on your system for IRXSAY WRITEERR output.

Note: For more information about the EYUTRACE stem variable, see Developing CICSPlex SM applications.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

VAR*(data-area)*
Names a variable that contains the resource table record to be parsed.

Chapter 3. CICSPlex SM API commands

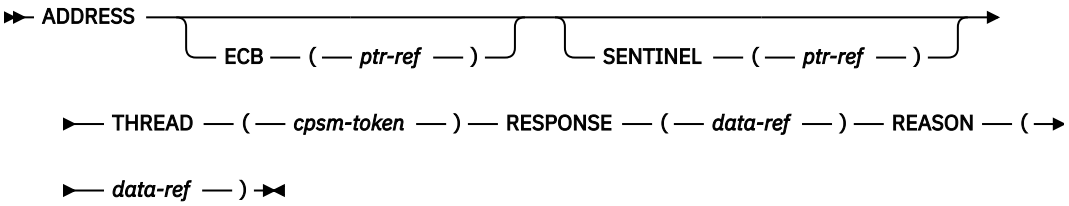
This section contains detailed descriptions of the API commands. All of these commands can be used with either the command-level interface or the REXX runtime interface.

Each description includes the following information, as appropriate:

- A description of the command
- Usage notes
- Related commands
- Syntax of the command
- Available options for the command
- Responses returned by the command

ADDRESS

Provide access to CICSPlex SM storage areas.



Description

The ADDRESS command provides access to CICSPlex SM storage areas.

- ADDRESS returns the addresses of two control fields that are associated with each API thread:
  - the event control block (ECB)
  - the sentinel.
- If your program is written in REXX, the ECB and sentinel values are returned as character representations of the hexadecimal addresses. You have to use the REXX STORAGE function to access the storage at those addresses.

Related commands

LISTEN, RECEIVE

Options

ECB*(ptr-ref)*
Names a variable to receive the address of the ECB that will be posted when asynchronous requests associated with this thread are awaiting processing. The ECB field is cleared whenever the counter value in the SENTINEL field reaches 0.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

SENTINEL*(ptr-ref)*
Names a variable to receive the address of a 4-byte counter of completed asynchronous requests associated with this thread.

The sentinel value increases each time an asynchronous request completes. Examples of asynchronous requests include:

- A command is issued with the NOWAIT option
- An event occurs that is named in a LISTEN command.

The sentinel value decreases when a RECEIVE command is issued. If the counter value is 0, it means there are no outstanding asynchronous requests to be received.

Note: Each API processing thread can handle a maximum of 256 asynchronous requests (as indicated by the SENTINEL counter) at one time.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the ADDRESS command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- ECB
- SENTINEL
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

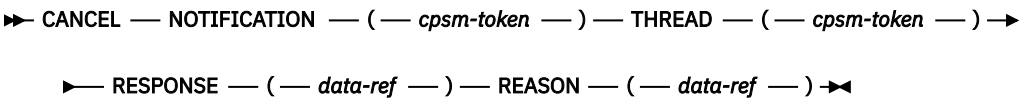
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

CANCEL

Cancel the notification request produced by a previous LISTEN command.



Description

This command cancels the notification request produced by a previous LISTEN command.

Related commands

LISTEN

Options

NOTIFICATION*(cpsm-token)*
Identifies the notification request to be cancelled. The *cpsm-token* value that identifies a notification request is returned by the LISTEN command.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the CANCEL command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- NOTIFICATION
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

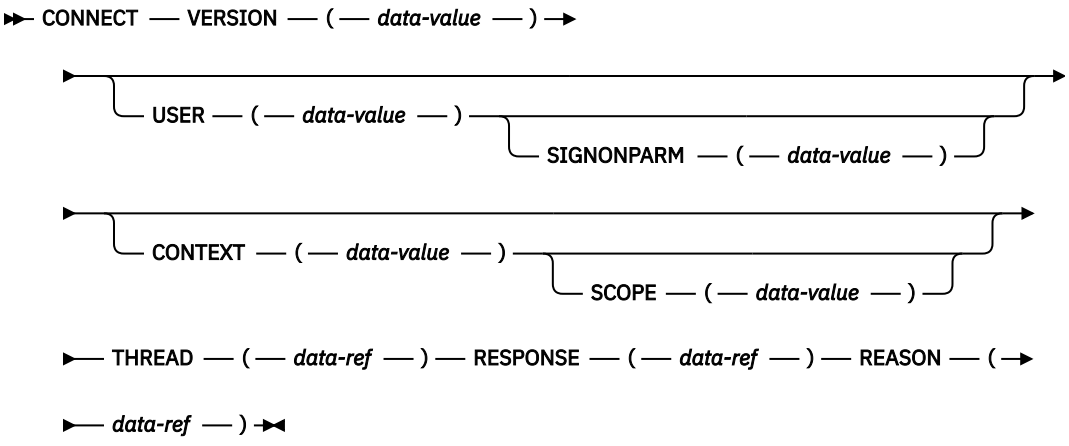
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

CONNECT

Establish a connection with CICSPlex SM, define an API processing thread, and provide default settings to be used by the thread.



Description

The specifics of the connection process depend upon the environment in which your program is running. For a complete description of the connection process, see Developing CICSPlex SM applications.

Related commands

DISCONNECT, QUALIFY, TERMINATE

Options

CONTEXT*(data-value)*
Identifies the default context for commands issued against this thread. The context must be the 1- to 8-character name of a CMAS or CICSplex.

The default context is in effect for all commands issued against the thread unless you override it for a specific command or change it by issuing the QUALIFY command. As an alternative to specifying a default context for the thread, you can specify the context for individual commands as they are processed.

If you do not specify the CONTEXT option, the default context for the thread is the CMAS to which the thread is connected.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

SCOPE*(data-value)*
Identifies the default scope for commands issued against this thread.

The SCOPE option qualifies the CONTEXT option. When the context is a CICSplex, the scope can be:

- The 1- to 8-character name of the CICSplex itself
- A CICS system or CICS system group within the CICSplex
- A logical scope, as defined in a CICSPlex SM resource description (RESDESC).

When the context is a CMAS, this option has no meaning and is ignored.

The default scope is in effect for all commands issued against the thread unless you override it for a specific command or change it by issuing the QUALIFY command. If you do not specify the SCOPE option, no default scope is assumed.

Note: Certain API commands require a valid scope when the context is a CICSplex. If you do not specify a scope on a CONNECT or QUALIFY command, you must specify the SCOPE option when you issue any of the following commands for a resource table that represents a CICS resource:

- GET
- PERFORM OBJECT
- PERFORM SET
- REFRESH
- SET

SIGNONPARM*(data-value)*
Identifies a 1- to 8-character sign-on parameter to be passed to the API security exit routine (EYU9XESV) at your enterprise. Password phrases are not supported.

If CMAS security is active and no security is defined in the environment where the API program is running, CICSPlex SM passes the USER and SIGNONPARM values from the CONNECT command to EYU9XESV. For more information about API security, see Developing CICSPlex SM applications.

THREAD*(data-ref)*
Names a variable to receive the fullword token that CICSPlex SM assigns to this processing thread.

This identifying token must be specified on all subsequent commands issued against this thread.

USER*(data-value)*
Identifies a 1- to 8-character user ID to be passed to the API security exit routine (EYU9XESV) at your enterprise.

If CMAS security is active and CICSPlex SM finds no security defined in the environment where the API program is running, it passes the USER and SIGNONPARM values from the CONNECT command to EYU9XESV. For more information about API security, see Developing CICSPlex SM applications.

VERSION*(data-value)*
Identifies the release of CICSPlex SM resource table data that you want to be available to your program. The VERSION value must be the 4-character number of a valid CICSPlex SM release, such as 0410 for CICS TS 4.1.

Notes:

1. The VERSION value must be 0120 or greater. The API cannot access data from a release of CICSPlex SM earlier than release 2.
2. The VERSION value must be less than or equal to the version of the CICSPlex SM runtime environment.
3. You can specify a VERSION value that is greater than the release under which your API program was originally written, provided:
   - You compile your program using the appropriate copy books for the version specified.
   - Your program is compatible with the copy books for the version specified.

   For complete details on things to consider when running under a different release, see Developing CICSPlex SM applications.

Conditions

The following is a list of the RESPONSE values that can be returned by the CONNECT command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abnormally terminated.

EXCEPTION
Command processing encountered an exceptional condition.

ENVIRONERROR
An environment error occurred for one of the following reasons:

APITASKERR
The API control subtask encountered an error during startup.

INVALIDTCB
A program using the application stub for non-CICS environments, EYU9ABSI, tried to connect when the program was executing on a CICS TCB.

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

SOERESOURCE
A required resource that is owned by the Environment Services System Services (ESSS) address space is not available.

SOLRESOURCE
A required resource that is locally owned (that is, owned by the address space where the processing thread is running) is not available.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- SCOPE
- SIGNONPARM
- USRID
- VERSION

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

CPSMSERVER
The CMAS to which the processing thread was trying to connect is not available.

CPSMSYSTEM
No CICSPlex SM systems are available.

CPSMVERSION
No CICSPlex SM system at the specified version is available.

NOTPERMIT
A not permitted condition occurred for one of the following reasons:

EXPIRED
The security authorization of the specified user ID has expired.

SIGNONPARM
The specified sign-on parameter is not authorized for the user ID.

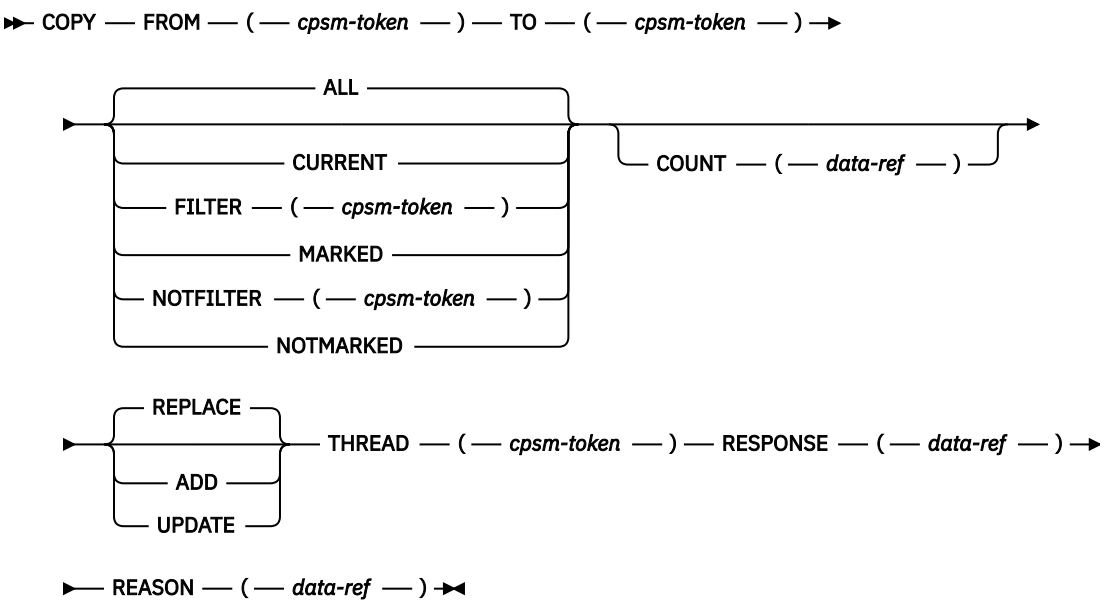
USRID
The specified user ID does not have the required security authorization.

VERSIONINVL
A version conflict occurred for the following reason:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

COPY

Copy resource table records.



Description

This command copies some or all of the resource table records in one result set to another result set on the same processin thread.

- The COPY command always begins processing with the last record that was fetched, rather than the next one in the result set.
- The target result set can be an existing result set or a new one that is created by this process. If you specify an existing result set as the target, you can either overwrite the existing records or add to them.
- A result set can contain only one record for a given resource. If duplicate records are found during the copy process, the ADD, REPLACE or UPDATE option you specified determines which record is retained.
- To copy selected records from a source result set, you can use:
  - The SPECIFY FILTER command to define a filter for the source result set.
  - The MARK and UNMARK commands to mark records in the source result set. Any marks you place on records in the source result set are not retained when those records are copied to the target result set.
- The relative position of records in the target result set may not be the same as it was in the source result set. The position can be affected by:
  - Deleted records being left in the source result set (when COPY ALL is specified) and other records assuming their position in the target result set.
  - The sort order associated with the target result set, if any. If the target result set does not exist, records are copied in the same order as they appeared in the source result set. If an existing result set is named as the target, records are copied and then sorted according to the sort order that was in effect for that result set.

Related commands

DELETE, DISCARD, GET, GETDEF, LOCATE, MARK, ORDER, PERFORM OBJECT, QUERY, SPECIFY FILTER

Options

ADD

Adds the resource table records from the source result set to an existing target result set. If duplicate records are found, the record in the target result set is retained.

If no existing result set is specified as the target, the ADD option is ignored.

ALL
Copies all the resource table records in the source result set to the target result set.

Any records that have been deleted from the source result set are not copied. In effect, the ALL option compresses a result set by leaving deleted records in the source result set and copying the remaining records to a new result set.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records in the target result set after the copy process is complete.

CURRENT
Copies only the current resource table record in the source result set to the target result set.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option copies only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

FROM*(cpsm-token)*
Identifies the source result set for this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- PERFORM OBJECT.

MARKED
Copies only those resource table records that are marked in the source result set. You can mark resource table records by using the MARK and UNMARK commands.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option copies only those resource table records that do not meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Copies only those resource table records that are not marked in the source result set. You can mark resource table records by using the MARK and UNMARK commands.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

REPLACE
Deletes the resource table records in an existing target result set and replaces them with the results of this copy operation. If the copy operation does not result in any resource table records being copied, the target result set is discarded.

If no existing result set is specified as the target, the REPLACE option is ignored.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TO*(cpsm-token)*
Identifies the target result set for this operation. The result set can be one produced by any of these commands:

- COPY

- GET
- GETDEF
- PERFORM OBJECT.

Note: The target result set cannot be the same as the source result set that you specified on the FROM option.

If this field is:

- Set to binary zero (in COBOL, C, PL/I or Assembler)
- An uninitialized variable (in REXX).

CICSPlex SM creates a new result set and returns its identifying token in the same field.

UPDATE
Updates an existing target result set with resource table records from the source result set. If duplicate records are found, the record in the source result set replaces the record in the target result set.

If no existing result set is specified as the target, the UPDATE option is ignored.

Conditions

The following is a list of the RESPONSE values that can be returned by the COPY command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria.

BUSY
A busy condition occurred for one of the following reasons:

FROM
The source result set specified on the FROM option is being processed by another command.

TO
The target result set specified on the TO option is being processed by another command. This condition can occur if you specified the same result set on the FROM and TO options.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INCOMPATIBLE
An incompatible condition occurred for one of the following reasons:

INVALIDOBJ
The target result set specified on the TO option is not compatible with the source result set specified on the FROM option. The result sets must contain the same type of resource table records.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- FILTER
- FROM
- NOTFILTER
- THREAD
- TO.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

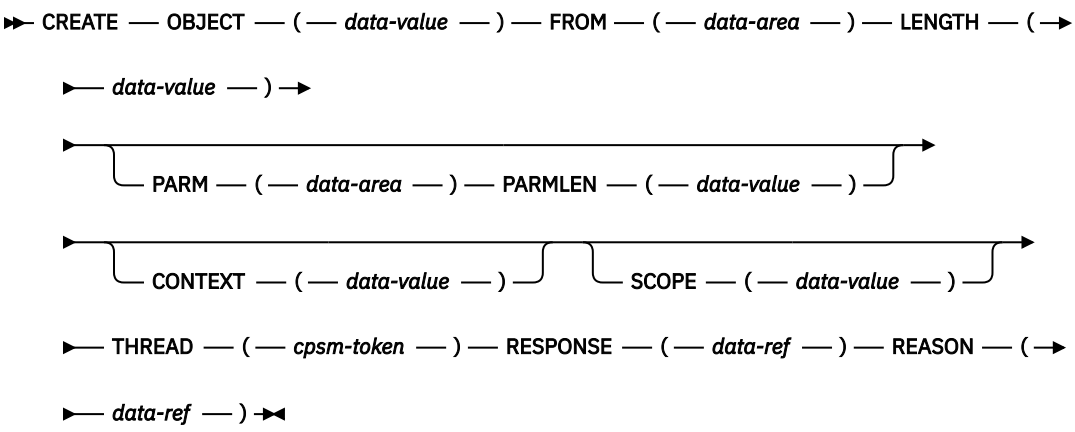
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

CREATE

Create a new CICSPlex SM or CICS definition.



Description

This command creates a new CICSPlex SM or CICS definition using the attribute values you specify. The new definition is stored in the CICSPlex SM data repository. For definitions that have a CICSplex

as their context (such as workload management or real-time analysis definitions), the new definition is automatically distributed to all the CMASs involved in managing the CICSplex.

Related commands

REMOVE, UPDATE

Options

CONTEXT*(data-value)*
Identifies the context for this command. The context must be the 1- to 8-character name of a CMAS or CICSplex.

If you do not specify the CONTEXT option, the default context for the thread is assumed.

FROM*(data-area)*
Identifies a buffer containing a resource table record that represents the definition to be created.

The record must include all of the attributes for the resource table specified on the OBJECT option. For optional attributes that you do not want to specify, set the field to null (that is, zero) values.

See CICSPlex SM resource tables for a list of all permitted null values.

LENGTH*(data-value)*
A fullword value that specifies the length of the FROM buffer.

OBJECT*(data-value)*
Identifies the resource table that represents the definition being created. This value must be the 1- to 8-character name of a valid CICSPlex SM definition or CICS definition resource table. For a list of the CICSPlex SM resource tables by type, see Developing CICSPlex SM applications.

PARM*(data-area)*
Identifies a buffer containing the parameter expression to be used in creating the definition.

For details on how to use a parameter expression with the CREATE command, see Developing CICSPlex SM applications. For a description of the parameters that are valid for a given resource table, see CICSPlex SM resource tables overview in Reference and CICSPlex SM resource tables overview in Reference.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

SCOPE*(data-value)*
Identifies the scope for this command.

If the current context (as set by this command or a previous CONNECT or QUALIFY command) is a CICSplex and the OBJECT option identifies a CICS definitional resource and the PARM option includes the CSD parameter, a valid scope can be specified.

The scope can be a CICS system within the CICSplex. If the current context is a CMAS or the OBJECT option identifies any other type of resource table, or the CSD parameter is not specified on a CICS definitional resource, this option has no meaning and is ignored.

If SCOPE applies to the command and you do not specify the SCOPE option, the default scope for the thread is assumed. If the current context is a CICSplex and no default scope has been set on a CONNECT or QUALIFY command, you receive an INVALIDPARM response for the SCOPE option.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the CREATE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
For CSD requests only, one of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs to which the request was directed did not respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- FROM
- LENGTH
- OBJECT
- PARM
- PARMLEN
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

CSDAPI
Support for the CICSPlex SM API to access the CICS CSD is not available.

MAINTPOINT
The maintenance point for the current context is not available.

NOTPERMIT
A not permitted condition occurred for one of the following reasons:

USRID
The user ID associated with the processing thread does not have the required security authorization.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for one of the following reasons:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if:

- The resource table is missing required attributes, contains one or more conflicting attributes, or is a duplicate.
- A CICS resource definition contains attributes that would cause the EXEC CICS CREATE command to issue warnings.

Use the FEEDBACK command to retrieve additional data about this error.

INVALIDATTR
One of the resource table attributes is invalid.

INVALIDVER
The specified version of the resource table is not supported by CICSPlex SM.

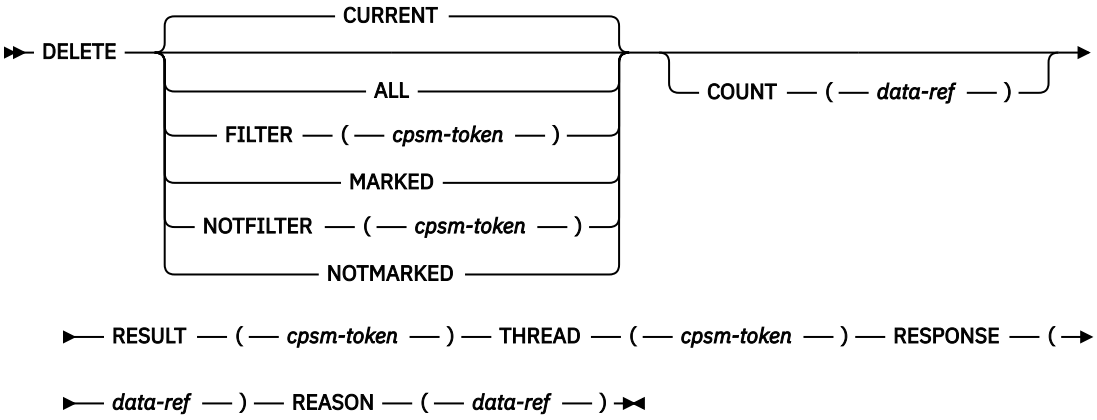
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

DELETE

Delete resource table records.



Description

This command deletes one or more resource table records from a result set.

- The DELETE command always begins processing with the last record that was fetched, rather than the next one in the result set.
- The records you delete are marked as deleted, but they retain their positions in the result set. The remaining records also retain their positions; they are not renumbered. Any API commands that you issue after a DELETE command skip over the deleted records in a result set. One exception is the ORDER command, which sorts all the records in a result set, including deleted records. If you try to issue a command against a deleted record, you receive a RESPONSE value of NODATA.
- To remove deleted records and compress a result set, you can copy the remaining records to a new result set. Use the COPY command with the ALL option to copy all the records in a result set except those that have been deleted.

  Note: Deleted records are also removed and the remaining records renumbered when you issue a REFRESH command.

Related commands

COPY, DISCARD, GET, GROUP, LOCATE, MARK, ORDER, PERFORM OBJECT, REFRESH, SPECIFY FILTER

Options

ALL
Deletes all the resource table records in the result set.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records in the result set after the delete process is complete.

CURRENT
Deletes only the current resource table record in the result set.

Note: The record pointer remains positioned on the deleted record. If you issue another API command with the CURRENT option before repositioning the pointer, you receive a RESPONSE value of NODATA.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option deletes only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

MARKED
Deletes only those resource table records that are marked in the result set. You can mark resource table records by using the MARK and UNMARK commands.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option deletes only those resource table records that do not meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Deletes only those resource table records that are not marked in the result set. You can mark resource table records by using the MARK and UNMARK commands.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GROUP
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the DELETE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

SOLRESOURCE
A required resource that is locally owned (that is, owned by the address space where the processing thread is running) is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- FILTER
- NOTFILTER
- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

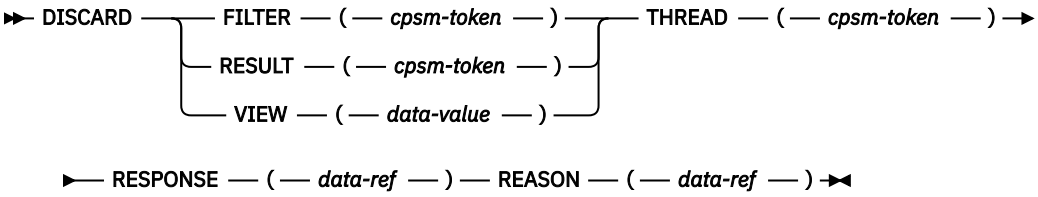
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

DISCARD

Discard a result set, filter, or view.



Description

This command discards a result set, filter, or view.

Related commands

COPY, GET, GETDEF, GROUP, PERFORM OBJECT, SPECIFY FILTER, SPECIFY VIEW

Options

FILTER*(cpsm-token)*
Identifies the filter to be discarded. The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be discarded. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- GROUP

- PERFORM OBJECT.

Note: If you discard a result set that was summarized by the GROUP command, all of the summarized result sets are also discarded.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

VIEW*(data-value)*
Identifies the view to be discarded. This value must be the 1- to 8-character name of a view as defined on a SPECIFY VIEW command.

Conditions

The following is a list of the RESPONSE values that can be returned by the DISCARD command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INUSE
An in use condition occurred for one of the following reasons:

FILTER
The specified filter is currently in use and cannot be discarded.

VIEW
The specified view is currently in use and cannot be discarded.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- FILTER
- RESULT
- THREAD
- VIEW.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

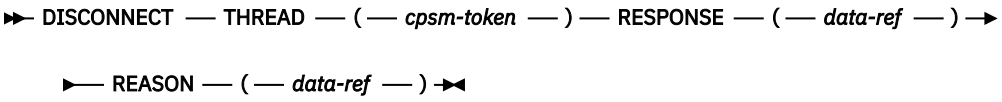
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

DISCONNECT

Disconnect an API processing thread from CICSPlex SM.



Description

Any resources that are associated with the thread are released, including result sets, filters, views, diagnostic data, and outstanding asynchronous requests.

Related commands

CONNECT, TERMINATE

Options

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be disconnected. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the DISCONNECT command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

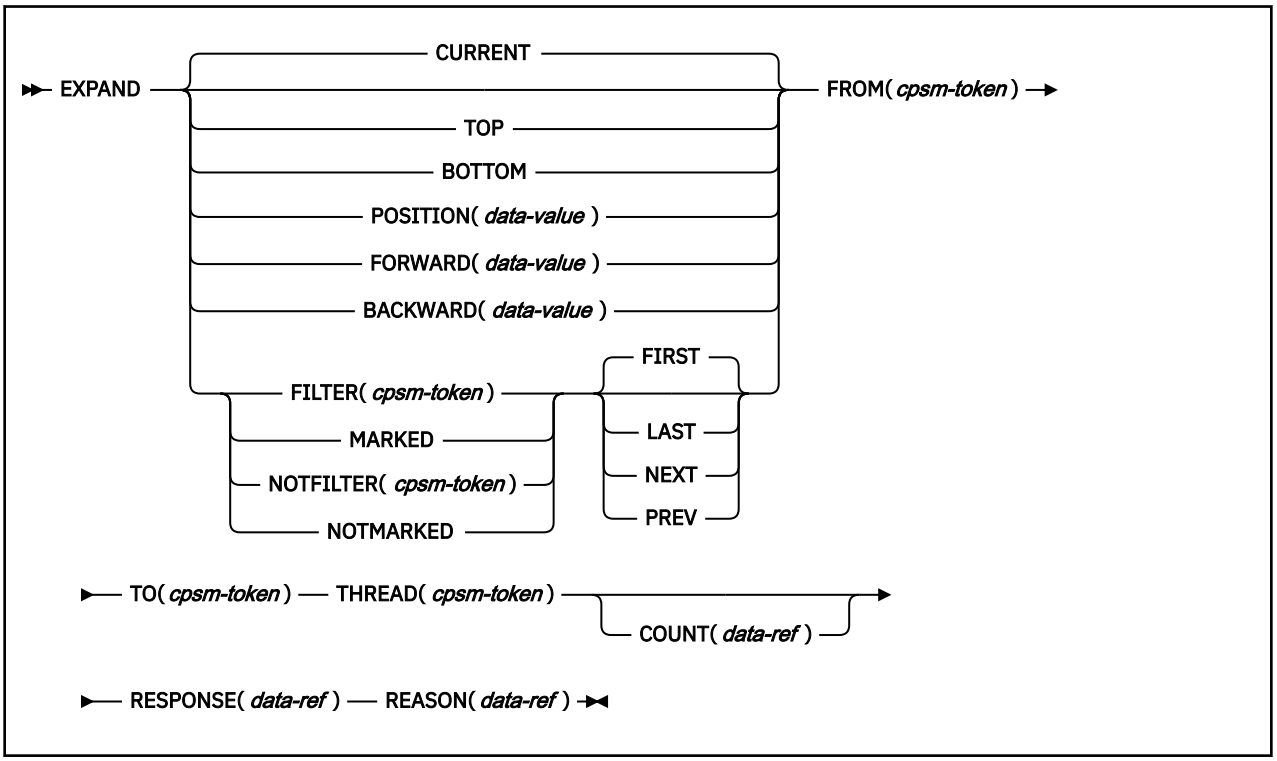
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

EXPAND

Return a result set containing all of the records summarized in a summary record.



Description

This command supports the expansion of summary result sets. The command accepts a token from a summarized result set produced by the GROUP command, and a selected record identified by the position of the record pointer in the result set to be expanded. The position of the record pointer depends on the options that you specify on the command. It creates a new result set that contains all the records that are summarized in a summary record.

Related commands

FETCH, GET, GROUP, LOCATE, MARK, ORDER, QUERY, REFRESH, SPECIFY FILTER, UNMARK

Options

BACKWARD*(data-value)*
Expands the record at the position arrived at by moving backwards from the current pointer position for *data-value*number of records. If the *data-value* value is greater than the remaining number of records, the first record in the summary result set is expanded.

BOTTOM
Expands the last record in the summary result set.

COUNT*(data-ref)*
The number of resource table records in the TO result set after this operation is complete. This parameter is output-only.

CURRENT
Expands the current record in the FROM result set.

FILTER*(cpsm-token)*
Identifies the filter to be used for this operation and performs an EXPAND operation on the record or records that match the filter criteria. It is used with the FIRST, LAST, NEXT and PREV options.

FIRST
Expands either the first marked record in the result set or the first record that matches the filter criteria. If no record is found, a NODATA code is returned.

FORWARD*(data-value)*
Expands the record at the position arrived at by moving forwards from the current pointer position for *data-value* number of records. If the *data-value* value is greater than the remaining number of records in the summary result set, the last record is expanded.

FROM *(cpsm-token)*
The summary result set on which the EXPAND command is to operate. If no matching result set can be found, an INVALIDPARM return code is issued with a reason code of FROM.

LAST
Expands either the last marked record in the result set or the last record that matches the filter criteria. If no record is found, a NODATA code is returned.

MARKED
Expands one or more records that have been selected using the MARK command. It is used in conjunction with the FIRST, LAST, NEXT, and PREV options.

You can mark resource table records by using the MARK and UNMARK commands.

NEXT
Starting at the record currently selected and moving forward through the result set, NEXT expands either the next marked record or the next record that matches the filter criteria. If no record is found, a NODATA code is returned.

NOTFILTER*(cpsm-token)*
Identifies the filter that is to be used for this operation and performs an EXPAND operation on the record or records that do not match the filter criteria. It is used in conjunction with the FIRST, LAST, NEXT and PREV options.

NOTMARKED
Expands one or more records that have been left unselected by the MARK command. It is used in conjunction with the FIRST, LAST, NEXT and PREV options.

You can mark resource table records by using the MARK and UNMARK commands.

POSITION*(data-value)*
Expands the record at a position in the summary result set indicated by the supplied value.

PREVIOUS
Starting at the record currently selected and moving backwards through the result set, PREVIOUS expands either the next marked record in the result set or the next record that matches the filter criteria. If no record is found, a NODATA code is returned.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
The API thread to be used for the EXPAND operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TO *(cpsm-token)*
Identifies the summary result set to contain the expanded records on which the EXPAND command operates. If this result set already exists, any existing resource table records that relate to it are replaced by the resource table records produced by this EXPAND command.

TOP
Expands the first record in the summary result set.

Conditions

The following RESPONSE values can be returned by the EXPAND command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified criteria, for one of the following reasons:

BACKWARD
No more records satisfy the search criteria in the backward direction.

FORWARD
There are no more records that satisfy the search criteria in the forward direction.

BUSY
A busy condition occurred for one of the following reasons:

FROM
The result set specified on the FROM option is being processed by another command.

TO
The result set specified on the TO option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing ended abnormally.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- BACKWARD
- FORWARD
- POSITION
- FILTER
- NOTFILTER
- FROM
- TO
- THREAD
- COUNT

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

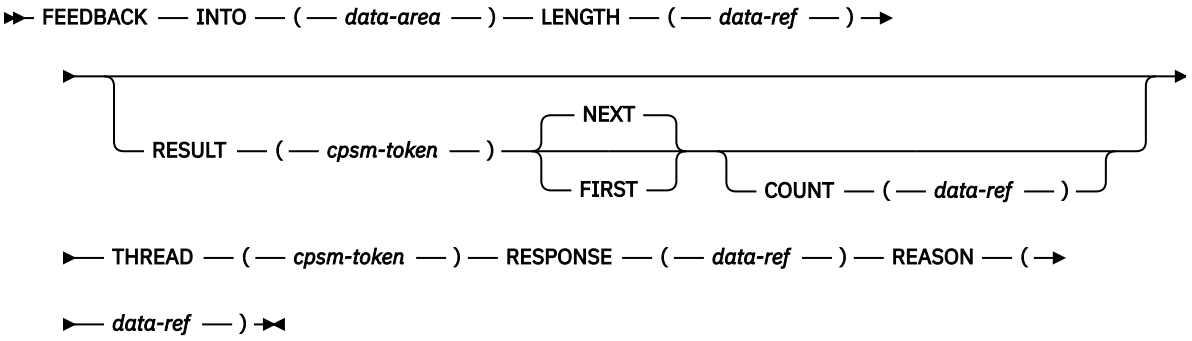
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

FEEDBACK

Retrieve diagnostic data.



Description

This command retrieves diagnostic data about a previously issued API command.

- The diagnostic data is returned as FEEDBACK resource table records.
- If the previous command involved processing a result set and it returned a RESPONSE value other than OK, a FEEDBACK resource table record is appended to the end of each resource table record in the result set that had an error associated with it causing the non-OK RESPONSE to be sent. The diagnostic data is available to the FEEDBACK command until another command processes the same result set. At that point, the data is replaced with FEEDBACK records for the subsequent command.

  Note: If a command that processed a result set returned a RESPONSE value of OK, FEEDBACK records are produced if CICS returns additional information in the EIBRESP2 field.
- If the previous command did not process a result set, the FEEDBACK resource table records are returned in a separate feedback area. The records in that feedback area are cleared and refreshed for each command that is not result set-oriented. So for commands that place their diagnostic data in the feedback area rather than in a result set, FEEDBACK can retrieve data only for the most recently issued command.
- Once you have issued the FEEDBACK command to retrieve diagnostic data for a command, the feedback record or area is cleared. You cannot request the same FEEDBACK resource table records more than once.
- If a command is processed asynchronously (that is, you specify the NOWAIT option) the diagnostic data for that command is returned in the ASYNCREQ notification resource table. No FEEDBACK resource table records are produced for an asynchronous request.
- Diagnostic data is not available for these commands:
  - DISCONNECT

  - FEEDBACK
  - TERMINATE
- The TBUILD and TPARSE commands supplied for use in REXX programs do not provide any useful FEEDBACK information.

For a complete description of the FEEDBACK resource table, see FEEDBACK Resource Table in Reference.

Options

COUNT*(data-ref)*
Specifies the number of feedback records to be retrieved from the result set named in the RESULT option. If you do not specify the COUNT option, only one feedback record is retrieved.

If you are retrieving multiple feedback records, they are placed one after another in the INTO buffer. The INTO buffer must be long enough to hold all the feedback records being retrieved.

The value that CICSPlex SM returns in this field depends on the RESPONSE value for the FEEDBACK command:

OK
The actual number of records returned in the INTO buffer.

WARNING AREATOOSMALL
The number of records returned in the INTO buffer, which is not the total number of records requested.

INVALIDPARM LENGTH
The field is not set because the INTO buffer was not long enough to hold even one resource table record.

FIRST
Retrieves the first feedback record from the result set named in the RESULT option.

If you specify the COUNT option, FIRST retrieves the specified number of records, beginning with the first record in the result set.

INTO*(data-area)*
Identifies a buffer (or stem variable, in REXX) to receive the feedback data. This buffer must be long enough to hold all the feedback data being retrieved.

LENGTH*(data-ref)*
A fullword value that specifies the length of the INTO buffer.

The value that CICSPlex SM returns in this field depends on the RESPONSE value for the FEEDBACK command:

OK
The actual length of the data returned in the INTO buffer.

WARNING AREATOOSMALL
The buffer length that would be required to hold all the requested records.

INVALIDPARM LENGTH
The field is not set because the INTO buffer was not long enough to hold even one resource table record.

NEXT
Retrieves the next available feedback record from the result set named in the RESULT option.

If you specify the COUNT option, NEXT retrieves the specified number of records, beginning with the next record in the result set.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies an API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GROUP
- PERFORM OBJECT.

Use the RESULT option to retrieve feedback data about a previously issued command that processed a result set. Use FEEDBACK without the RESULT option to retrieve data about the most recently issued command that did not process a result set.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the FEEDBACK command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria, or a command that processed a result set returned a RESPONSE of OK.

WARNING
The command completed processing with a warning, for the following reason:

AREATOOSMALL
The INTO buffer is not long enough to hold the number of records requested and available.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- COUNT
- INTO
- LENGTH

- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

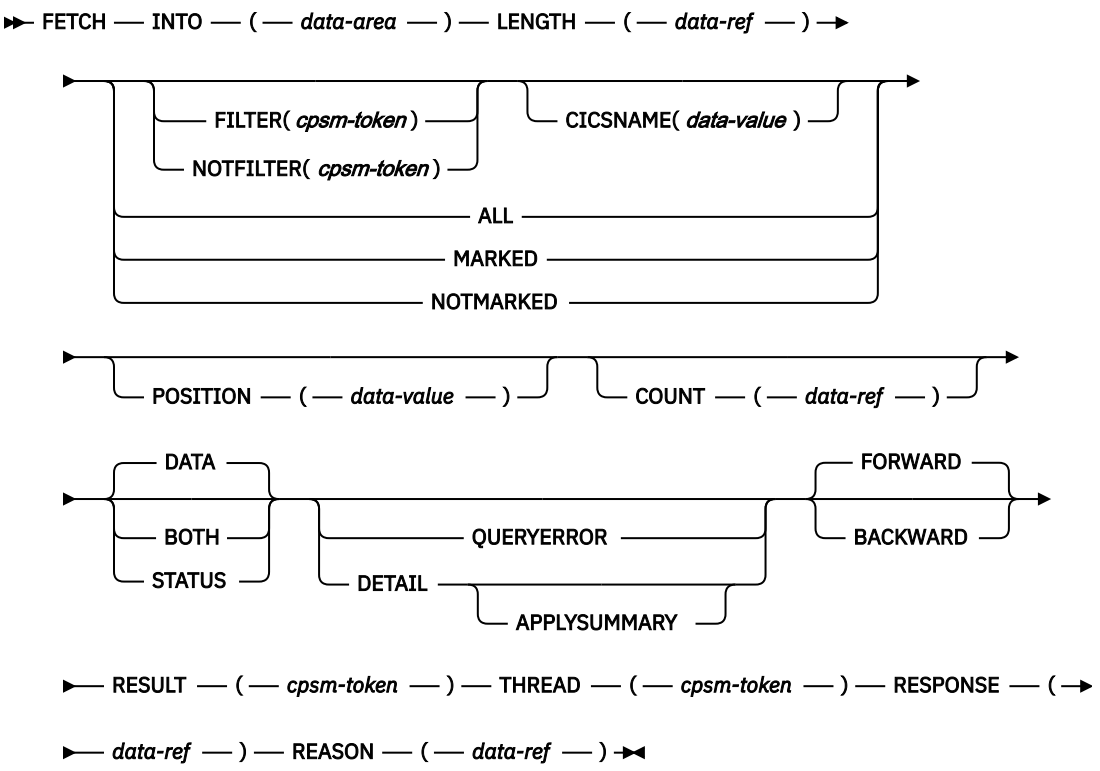
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

FETCH

Retrieve data and status information for resource table records.



Description

This command retrieves data and status information for one or more resource table records in a result set.

- After a FETCH command, the record pointer is usually positioned to the next record in the result set (that is, the record following the last record fetched in whichever direction the pointer was moving, forward or backward). However, the following API commands always act upon the last record that was fetched (that is, the record pointer is not advanced):
  - COPY
  - DELETE
  - MARK
  - UNMARK
  - PERFORM SET CURRENT
  - REFRESH CURRENT
  - SET CURRENT
- If no records were fetched (because no records matched the specified criteria), the pointer is positioned to the top or bottom of the result set, depending on which direction it was moving.

Related commands

COPY, GET, GETDEF, GROUP, LOCATE, MARK, ORDER, PERFORM OBJECT, QUERY, SPECIFY FILTER

Options

ALL
Retrieves all the resource table records in the result set. When you specify ALL, the POSITION and COUNT options are ignored.

APPLYSUMMARY
Apply any, some, or all of the following options to the summary records and retrieve the detail records associated with the summary records selected.

- MARKED
- NOTMARKED
- FILTER
- NOTFILTER

The APPLYSUMMARY option is only valid if the DETAIL option is also specified.

If the DETAIL option is specified without the APPLYSUMMARY option the result will be as described under the DETAIL option.

If neither the DETAIL option nor the APPLYSUMMARY option are specified but any combination of some or all of the following record selection options: MARKED, NOTMARKED, FILTER, and NOTFILTER are issued against a summary result set, the record selection options are applied to the summary result set and the selected summary records are retrieved.

BACKWARD
Begins the retrieval process with the last record fetched and continues in a backward direction through the specified result set.

BOTH
Retrieves both the resource table data and the OBJSTAT status information about the last action performed against the resource table. Each record contains OBJSTAT information followed by resource table data.

CICSNAME*(data-value)*
Specifies a 1- to 8-character specific or generic CICS system name to be used for this operation.

The CICSNAME option indicates that only those resource table records that originate from CICS systems that match the specified name pattern should be considered for retrieval. When CICSNAME is specified in conjunction with FILTER or NOTFILTER, only records which meet the FILTER or

NOTFILTER requirements and also match the CICSNAME pattern will be considered. The number of records retrieved is determined by the COUNT option.

When you specify CICSNAME, the result set named on the RESULT option must not be a summarized result set and must contain resource table records that have an EYU_CICSNAME attribute. If the result set specifed by RESULT contains summarized records or resource table records that do not have an EYU_CICSNAME attribute, you receive an INVALIDPARM response for the CICSNAME option.

COUNT*(data-ref)*
Specifies the number of resource table records to be processed.

The COUNT option applies to the result set named in the RESULT option. When you also specify the DETAIL option, COUNT provides the number of summary records in the summarized result set in RESULT for which source records are returned. The OBJSTAT table for each summary record contains the number of source records that will be returned for that record if the DETAIL option is specified.

If you do not specify the COUNT option, the default is one.

If the COUNT option is specified, COUNT contains the number of records processed. In most cases this is also the number of records returned. However, if you also specify the DETAIL option, all source records associated with the requested number of summary record are retrieved. This is normally greater than the number specified in the COUNT option.

If you are retrieving multiple records, they are placed one after another in the INTO buffer. The INTO buffer should be long enough to hold all the records being retrieved.

The value that CICSPlex SM returns in this field depends on the RESPONSE value for the FETCH command as follows:

OK
The actual number of records returned in the INTO buffer.

WARNING AREATOOSMALL
The number of records returned in the INTO buffer, which is not the total number of records requested.

INVALIDPARM LENGTH
The field is not set because the INTO buffer was not long enough to hold even one resource table record.

DATA
Retrieves only the specified resource table data. The records do not contain any OBJSTAT status information about the last action performed against the resource table.

Note: The OBJSTAT information includes a summary count field that is set when resource table records are summarized using the GROUP command. If you plan to GROUP the resource table records and you want to know how many records are combined to form a summary record, you should specify BOTH to obtain both data and OBJSTAT information when the records are fetched.

DETAIL
Retrieves the source records associated with specific summary resource table records.

When you specify DETAIL, the result set named in the RESULT option must be a summarized result set. DETAIL expands the summary record by retrieving the resource table records associated with it from the source result set. If you do not specify DETAIL when a summarized result set is being processed, the summary records themselves are retrieved. If the result set is not a summarized result set, this option has no meaning and is ignored.

You can use the FORWARD or BACKWARD options along with DETAIL to select which summary record you want to expand. The FORWARD and BACKWARD options also control the direction in which records are retrieve from the source result set.

By default, all the source records associated with the summary record or records are retrieved. However, you can use the FILTER or NOTFILTER option to limit the records retrieved from the source result set. You can also use the MARKED or NOTMARKED option to retrieve only those records associated with the summary record that are marked (or not marked) in the source result set.

You cannot explicitly position the record pointer in the source result set. When you specify DETAIL, the POSITION option refers to the record in the summary result set. If the APPLYSUMMARY option is specified, FILTER, NOTFILTER, MARKED, and NOTMARKED options are applied to records in the summary result set rather than to the source records.

For more information on processing summarized result sets, see Developing CICSPlex SM applications. For a description of the GROUP command, which creates summarized result sets, see "GROUP" on page 60.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option indicates that only those resource table records that meet the specified filter criteria should be considered for retrieval. The number of records that are retrieved is determined by the COUNT option.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

FORWARD
Begins the retrieval process with the next record (that is, the record that follows the last record fetched) and continues in a forward direction through the specified result set.

INTO*(data-area)*
Identifies a buffer (or stem variable, in REXX) to receive the resource table records. This buffer must be long enough to hold all the records being retrieved.

LENGTH*(data-ref)*
A fullword value that specifies the length of the INTO buffer.

The value that CICSPlex SM returns in this field depends on the RESPONSE value for the FETCH command:

OK
The actual length of the data returned in the INTO buffer.

NODATA
The length is set to zero.

WARNING AREATOOSMALL
The buffer length that would be required to hold all the requested records.

INVALIDPARM LENGTH
The field is not set because the INTO buffer was not long enough to hold even one resource table record.

MARKED
Indicates that only those resource table records that are marked in the result set should be considered for retrieval. The number of records that are retrieved is determined by the COUNT option.

You can mark resource table records by using the MARK and UNMARK commands.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option indicates that only those resource table records that do not meet the specified filter criteria should be considered for retrieval. The number of records that are retrieved is determined by the COUNT option.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Indicates that only those resource table records that are not marked in the result set should be considered for retrieval. The number of records that are retrieved is determined by the COUNT option.

You can mark resource table records by using the MARK and UNMARK commands.

POSITION*(data-value)*
Begins the retrieval process with the nth resource table record in the result set.

This value must be a number that identifies the record's relative position in the result set. The first record in a result set is identified by the number 1.

For example, to begin the retrieval process with the fifth resource table record in a result set, you would specify `POSITION(5)`.

Note: When the POSITION option is used with the DETAIL option to retrieve source records for a specific summarized result set record, the value of the COUNT option is forced to one (1). In this case, the value returned by the COUNT option is the number of source records summarized in the specified result set record.

QUERYERROR
Indicates that this request is to return MASQRYER resources generated by the last GET, PERFORM, or SET command to act on the result set.

Note: The data selection options FILTER, NOFILTER, MARKED, NOTMARKED, and CICSNAME are ignored if you specify the QUERYERROR option.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- GROUP
- PERFORM OBJECT.

STATUS
Retrieves only the OBJSTAT status information for the last action performed against the resource table. The records do not contain any resource table data.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the FETCH command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria, for one of the following reasons:

BACKWARD
There are no more records that satisfy the search criteria in the backward direction.

FORWARD
There are no more records that satisfy the search criteria in the forward direction.

WARNING
The command completed processing with a warning, for the following reason:

AREATOOSMALL
The INTO buffer is not long enough to hold the number of records requested and available.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

SOLRESOURCE
A required resource that is locally owned (that is, owned by the address space where the processing thread is running) is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- COUNT
- FILTER
- INTO
- LENGTH
- NOTFILTER
- POSITION
- RESULT
- THREAD
- CICSNAME

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

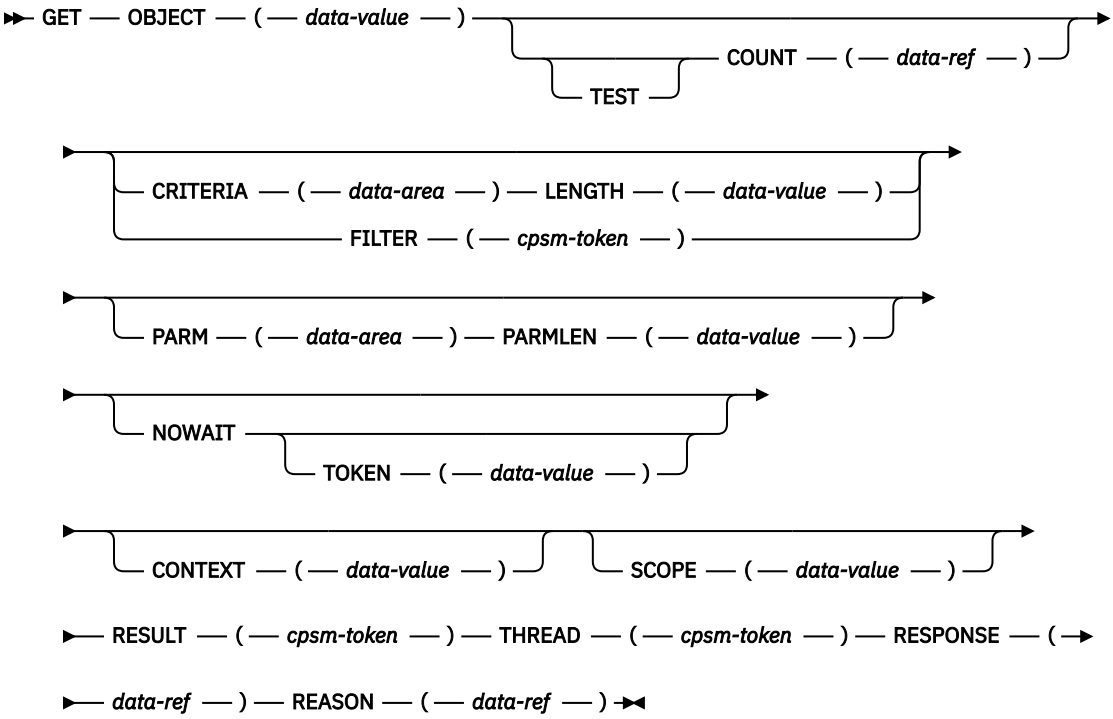
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

GET

Identify and optionally return a result set containing selected resource table records.



Description

This command identifies a result set containing selected resource table records. The command returns the identified records, or a count of the identified records, or both the records and the count.

- The resource table can be one that represents a CICS resource, a CICSPlex SM or CICS definition, or a CICSPlex SM run-time object.
- After a GET command, the record pointer is positioned to the top of the result set (that is, the first record in the result set).
- If the context and scope in effect when you issue a GET command include CICS systems that do not support the requested resource table, the request is ignored for those CICS systems.
- In some CICS environments, the resource table attribute values that are returned by CICSPlex SM for:

Resource table	Attribute value	CICS Environment
LOCTRAN	RESSEC(RESSECEXT)	CICS/MVS

do not match the CVDA values returned by CICS. The values returned by CICS conflict with CVDA values in other CICS environments. In order to retain the attributes' uniqueness, CICSPlex SM adds 9000 to the values returned by CICS. For more information about translating CVDA values, see TRANSLATE command.

Related commands

DISCARD, FETCH, GETDEF, QUERY, RECEIVE, REFRESH, SPECIFY FILTER, SPECIFY VIEW

Options

CONTEXT*(data-value)*
Identifies the context for this command. The context must be the 1- to 8-character name of a CMAS or CICSplex.

If you do not specify the CONTEXT option, the default context for the thread is assumed.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records in the target result set after this operation is complete.

CRITERIA*(data-area)*
Identifies a buffer containing the filter expression to be used for this operation. The CRITERIA option retrieves only those resource table records that meet the specified filter criteria.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option retrieves only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

LENGTH*(data-value)*
A fullword value that specifies the length of the CRITERIA buffer.

Note: The buffer length you specify should not include any data other than a filter expression.

NOWAIT
Returns control to your program as soon as the GET command has been accepted, which allows the command to be processed asynchronously.

If you specify the NOWAIT option, you must use a subsequent RECEIVE command to test for the completion of this request. The results of an asynchronous request are returned as ASYNCREQ resource table records.

Note: If you specify the TOKEN option, the NOWAIT option is assumed by default.

OBJECT*(data-value)*
Identifies the resource table for which records are to be retrieved. This value must be the 1- to 8-character name of either a valid resource table or a valid view. If you are using the TEST parameter, you must specify a CICS resource table name for **OBJECT**. Specifying any other value, including a view name, results in an error.

PARM*(data-area)*
Identifies a buffer containing the parameter expression to be used in preselecting resource table records.

For details on how to use a parameter expression with the GET command, see Using the PARM option. For a description of the parameters that can be specified for a given resource table, see CICSPlex SM resource tables.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GROUP

- PERFORM OBJECT.

CICSPlex SM replaces the contents of the existing result set with the resource table records requested by this GET command.

If this field is:

- Set to binary zero (in COBOL, C, PL/I or Assembler)
- An uninitialized variable (in REXX).

CICSPlex SM creates a new result set and returns its identifying token in the same field.

If you are using the TEST parameter, you must specify a zero value for **RESULT**. Specifying any other value results in an error.

SCOPE*(data-value)*
Identifies the scope for this command.

If the current context (as set by this command or a previous CONNECT or QUALIFY command) is a CICSplex and the OBJECT option identifies a CICS resource, a valid scope is required. The scope can be:

- The 1- to 8-character name of the CICSplex itself
- A CICS system or CICS system group in the CICSplex
- A logical scope, as defined in a CICSPlex SM resource description (RESDESC).

If OBJECT identifies a CICS definitional resource and the PARM option includes the CSDGROUP parameter, a valid scope can be specified. The scope can be:

- A CICS system in the CICSplex.

If the current context is a CMAS or the OBJECT option identifies any other type of resource table this option has no meaning and is ignored.

If you do not specify the SCOPE option, the default scope for the thread is assumed. If the current context is a CICSplex and no default scope has been set on a CONNECT or QUALIFY command, you receive an INVALIDPARM response for the SCOPE option.

TEST
Returns only a count of the records that match the request. If you specify TEST, the request returns no records. If you specify TEST, you must specify a CICS resource name for the OBJECT parameter, and a zero value for the RESULT parameter.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOKEN*(data-value)*
Defines a 1- to 4-character token that you choose to correlate an asynchronous GET request with the result of a subsequent RECEIVE command. This token is for use by your program; CICSPlex SM makes no use of the value. The token is returned by the RECEIVE command when this GET request is complete.

Conditions

The following is a list of the RESPONSE values that can be returned by the GET command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

SCHEDULED
The command has been scheduled for processing.

NODATA
No records were found that matched the specified search criteria.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
One of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs or MASs to which the request was directed did not respond.

If only some of the CMASs or MASs did not respond, the GET command can still yield a valid result. COUNT might be greater than zero and RESULT might be non-zero. Such a result set contains data from those CMASs and MASs that did respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDATA
Invalid data was detected. The parameter that contains invalid data is returned as the reason value:

CRITERIA
An attribute value listed in the CRITERIA buffer is not valid for the specified attribute.

INVALIDCMD
The command is invalid for one of the following reasons:

FILTER
The filter expression passed on the operation is too large or complex.

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- CRITERIA
- FILTER
- LENGTH
- OBJECT
- PARM
- PARMLEN
- RESULT
- SCOPE

- TOKEN
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

MAINTPOINT
The maintenance point for the current context is not available.

SCOPE
Either none of the MASs in the specified scope are available or none of them support the requested resource table.

WORKLOAD
The workload identified on the API request is not available on the local CMAS.

NOTFOUND
A not found condition occurred for the following reason:

ATTRIBUTE
An attribute specified in the CRITERIA buffer was not found for the specified resource table.

NOTPERMIT
A not permitted condition occurred for the following reason:

USRID
The user ID associated with the processing thread does not have the required security authorization.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for the following reason:

DATAERROR
The value associated with one or more resource table attributes is invalid. Use the FEEDBACK command to retrieve additional data about this error.

VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

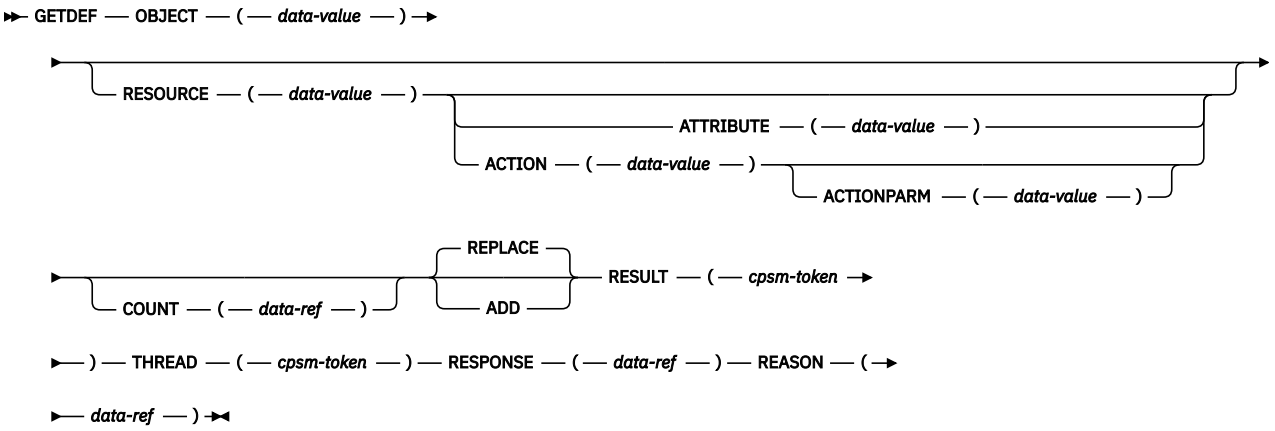
WARNING
The command completed processing with a warning, for the following reason:

MAXRECORDS
The number of records added to the result set by a MAS would have exceeded the MAXHISTRECS value for that MAS. Records within the MAXHISTRECS limit have been added to the result set. Modify the FILTER or PARM parameter values to increase or reduce the number of records the MAS should add to the result set.

Note: If a scope is specified that contains more than one MAS, the total number of records collected can exceed the MASHISTRECS value for an individual MAS.

GETDEF

Return a result set containing selected descriptive records for a resource table.



Description

This command returns a result set containing selected descriptive records for a resource table.

- GETDEF is a variation of the GET command. GET retrieves data records for the resource represented by a table. GETDEF, on the other hand, retrieves internal data that describes the resource table itself.
- The GETDEF command retrieves its data, which is called meta-data, from internal resource tables that describe each of the external resource tables. These internal resource tables are called CICSPlex SM metadata tables. The attributes of a CICSPlex SM metadata table are the characteristics of the external table, not the resource that it represents. For a list of the CICSPlex SM metadata resource tables that can be retrieved by GETDEF, see the description of the OBJECT option, Object.
- You can use GETDEF to find out what resource tables are available for processing by other commands. In addition, you can identify the attributes of a resource table, the values allowed for its modifiable attributes, and the actions that can be performed on it. You can also use GETDEF to request descriptions of the CICSPlex SM metadata resource tables themselves.
- You can use the GETDEF command only with resource tables supplied by CICSPlex SM. GETDEF is not valid for user-defined views of a resource table that were created by the SPECIFY VIEW command.
- You cannot use the REFRESH command to refresh the data records retrieved by GETDEF.

Related commands

DISCARD, FETCH, GET, LOCATE, QUERY

Options

ACTION*(data-value)*
The 12-character name of an action against the resource table for which CICSPlex SM metadata records are to be retrieved.

ACTIONPARM*(data-value)*
The 12-character name of a parameter to an action against the resource table for which CICSPlex SM metadata records are to be retrieved, as it appears in the API parameter string.

ADD
Adds the CICSPlex SM metadata resource table records that are being retrieved to an existing target result set. If no existing result set is specified as the target, the ADD option is ignored.

ATTRIBUTE*(data-value)*
Identifies one or more attributes of the resource table specified on the RESOURCE option for which CICSPlex SM metadata records are to be retrieved.

Depending on which CICSPlex SM metadata table is named in the OBJECT option, this value can be the 1- to 12-character name of a specific attribute or an asterisk (*), for all attributes in the resource table. If you do not specify the ATTRIBUTE option for an OBJECT that does not require it, data is retrieved for all attributes in the resource table.

For details on the CICSPlex SM metadata resource tables and the valid ATTRIBUTE values for each, see the description of the OBJECT option.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records in the target result set after this operation is complete.

OBJECT*(data-value)*
Identifies the type of meta-data to be retrieved for the resource table specified on the RESOURCE option. This value must be one of the following CICSPlex SM metadata resource table names:

OBJECT
One record is returned for each instance of the resource table specified on the RESOURCE option. The record describes the resource table's general characteristics. Related options and restrictions include:

- ACTION is ignored.
- ATTRIBUTE is ignored.
- ACTIONPARM is ignored.
- RESOURCE must be a specific resource table name or * for all resource tables.

OBJACT
One record is returned for each action that is available for the resource table specified on the RESOURCE option.

Related options and restrictions include:

- ACTION is normally omitted. If present, it may specify the name of an action or *.
- ATTRIBUTE is ignored.
- ACTIONPARM is ignored.
- RESOURCE must be a specific resource table name; a value of * is not allowed.

METADESC
One record is returned for each attribute of the resource table specified on the RESOURCE option. Each record provides only the basic structure of the attribute, including the name, data type, length, and offset in the resource table. Such information might be useful for accessing the attribute fields in a buffer returned by the FETCH command.

Related options and restrictions include:

- ACTION is ignored.
- ATTRIBUTE can be a specific attribute name or * for all attributes in the resource table.
- ACTIONPARM is ignored.
- RESOURCE must be a specific resource table name; a value of * is not allowed.

ATTR
One record is returned for each attribute of the resource table specified on the RESOURCE option. Each record provides complete information about the attribute.

Related options and restrictions include:

- ATTRIBUTE can be a specific attribute name or * for all attributes in the resource table.
- RESOURCE must be a specific resource table name; a value of * is not allowed.

ATTRAVA
One record is returned for each of the EYUDA or CVDA values that are valid for the specified attribute.

Related options and restrictions include:

- ATTRIBUTE must be the name of a specific attribute that has a data type of EYUDA, CVDAS, or CVDAT.
- RESOURCE must be a specific resource table name; a value of * is not allowed.

Note: The AVAAVAIL attribute of the ATTR internal resource table indicates whether an AVA list is available for a given attribute.

METANAME
One record is returned for each CVDA and EYUDA. The RESOURCE, ATTRIBUTE, ACTION and ACTIONPARM keywords are ignored.

METAPARM
One record is returned for every parameter for the specified RESOURCE and ACTION.

- ACTION must be a specific action name; a value of * is not allowed.
- ACTIONPARM is ignored.
- RESOURCE must be a specific table; a value of * is not allowed.

PARMAVA
One record is returned for the specified RESOURCE, ACTION, and ACTIONPARM.

- ACTION must be a specific action name; a value of * is not allowed.
- ACTIONPARM must be a specific parameter name; a value of * is not allowed.
- RESOURCE must be a specific table; a value of * is not allowed.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

REPLACE
Deletes the contents of an existing target result set and replaces them with the results of this operation. If the operation does not result in any CICSPlex SM metadata resource table records being selected, the target result set is discarded.

If no existing result set is specified as the target, the REPLACE option is ignored.

RESOURCE*(data-value)*
Identifies the resource table for which CICSPlex SM metadata records are to be retrieved.

If you specify the ATTRIBUTE option, this value must be the 1- to 8-character name of a specific CICSPlex SM resource table. Otherwise, you can specify a value of asterisk (*) to retrieve data for all resource tables.

Note: You can use GETDEF only with resource tables supplied by CICSPlex SM. GETDEF is not valid for user-defined views of a resource table that were created by the SPECIFY VIEW command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET

- GROUP
- PERFORM OBJECT.

CICSPlex SM replaces the contents of the existing result set with the resource table records requested by this GETDEF command. If the operation does not result in any resource table records being selected, the target result set is discarded.

If this field is:

- Set to binary zero (in COBOL, C, PL/I or Assembler)
- An uninitialized variable (in REXX).

CICSPlex SM creates a new result set and returns its identifying token in the same field.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the GETDEF command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
OBJACT was specified with the OBJECT option, but there are no actions defined for the specified RESOURCE.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INCOMPATIBLE
An incompatible condition occurred for the following reason:

INVALIDOBJ
The target result set specified on the RESULT option is not compatible with the output of this command. The result set must contain the same type of meta-data (as specified on the OBJECT option) as the command produces.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- ACTION
- ACTIONPARM
- ATTRIBUTE
- OBJECT
- RESOURCE
- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for one of the following reasons:

DATAERROR
The value associated with one or more resource table attributes is invalid. Use the FEEDBACK command to retrieve additional data about this error.

INVALIDVER
The specified version of the resource table is not supported by CICSPlex SM.

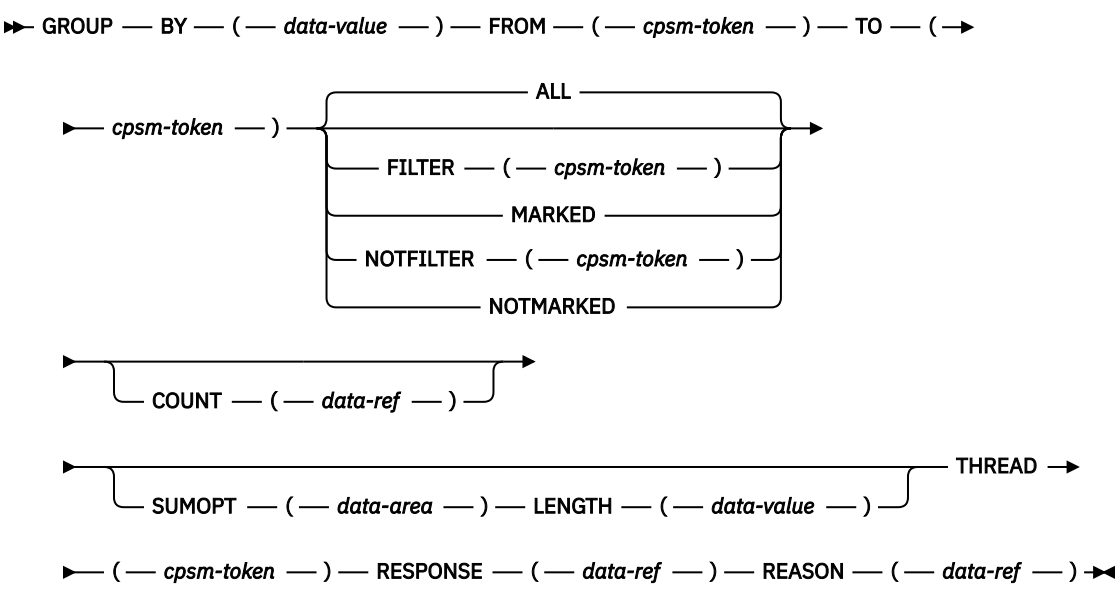
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

GROUP

Return a summarized result set.



Description

This command returns a summarized result set by grouping some or all of the resource table records in a result set.

- The target result set can be an existing result set or a new one that is created by this process. If you specify an existing result set as the target of a GROUP command:
  - It must be a summarized result set that was produced by a previous GROUP command against the same source result set.
  - It must contain the same type of resource table records currently found in the source result set.
  - The existing records in the result set are overwritten.
- To create a summarized result set from selected records of a source result set, you can use:
  - The SPECIFY FILTER command to define a filter for the source result set.
  - The MARK and UNMARK commands to mark records in the source result set.
- The GROUP command may be used only for attributes with a length of 251 or less. A RESPONSE(INVALIDPARM) REASON(BY) error occurs for attribute lengths greater than 251.
- For more information on processing summarized result sets, see Developing CICSPlex SM applications.

Related commands

DISCARD, EXPAND, FETCH, GET, LOCATE, MARK, ORDER, QUERY, SPECIFY FILTER

Options

ALL
Summarizes all the resource table records in the source result set.

BY*(data-value)*
Identifies the resource table attribute whose value is to be used as the grouping factor for this operation. This value must be the 1- to 12-character name of a valid attribute for the resource table.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records in the target result set after this operation is complete.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option summarizes only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

FROM*(cpsm-token)*
Identifies the source result set for this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- PERFORM OBJECT.

Note: If you discard the source result set, all of the summarized result sets that were created from it are also discarded.

LENGTH*(data-value)*
A fullword value that specifies the length of the SUMOPT buffer.

Note: The buffer length you specify should not include any data other than a summary expression.

MARKED
Summarizes only those resource table records that are marked in the source result set. You can mark resource table records by using the MARK and UNMARK commands.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option summarizes only those resource table records that do not meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Summarizes only those resource table records that are not marked in the source result set. You can mark resource table records by using the MARK and UNMARK commands.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

SUMOPT*(data-area)*
Identifies a buffer containing the summary expression to be used for this operation. The SUMOPT value overrides the default summary options for the resource table attributes.

For details on how to form a summary expression, see Developing CICSPlex SM applications. For a list of the default summary options for a given resource table, see the CICSPlex SM resource tables in Reference.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TO*(cpsm-token)*
Identifies the target result set for this operation.

If this field is:

- Set to binary zero (in COBOL, C, PL/I or Assembler)
- An uninitialized variable (in REXX).

CICSPlex SM creates a new summarized result set and returns its identifying token in the same field.

Otherwise, you can specify an existing summarized result that was produced by a previous GROUP command against the result set specified in the FROM option. That is, you can reuse a summarized result set, but only to resummarize the records in the same result set.

Note: If you specify the token of a previously produced summarized result set, make sure the result set still exists. When you discard a source result set, all of the summarized result sets that were created from it are also discarded.

Conditions

The following is a list of the RESPONSE values that can be returned by the GROUP command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria.

BUSY
A busy condition occurred for one of the following reasons:

FROM
The source result set specified on the FROM option is being processed by another command.

TO
The target result set specified on the TO option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- BY
- FILTER
- FROM
- LENGTH
- NOTFILTER
- SUMOPT
- THREAD
- TO.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

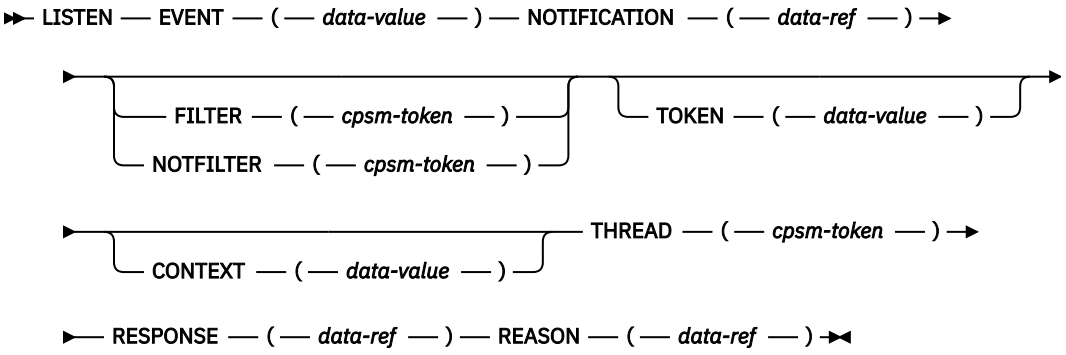
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

LISTEN

Request a notification be sent to the processing thread.



Description

This command requests that a notification be sent to the processing thread when a specific event occurs in the CICSplex.

- An event is represented by a resource table with a type of CPSM Notification.
- The LISTEN command is used in conjunction with the RECEIVE command. If you use LISTEN to request notification of an event, you must use a subsequent RECEIVE command to retrieve information about the event.
- An API processing thread can have a maximum of 256 completed asynchronous requests outstanding at one time. If you do not issue the RECEIVE command at regular intervals and your processing thread reaches its maximum of 256, asynchronous requests are discarded and are not processed. For a complete description of asynchronous processing, see Developing CICSPlex SM applications.

Related commands

ADDRESS, CANCEL, RECEIVE

Options

CONTEXT*(data-value)*
Identifies the context for this command. The context must be the 1- to 8-character name of a CMAS or CICSplex.

If you do not specify the CONTEXT option, the default context for the thread is assumed.

EVENT*(data-value)*
Identifies the resource table that represents the event to be listened for. This value must be the 1- to 8-character name of a valid CICSPlex SM Notification resource table. For a list of the CICSPlex SM resource tables by type, see Developing CICSPlex SM applications.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option listens for only those events that meet the specified filter criteria.

Using the FILTER option, you can limit the notifications you receive to events that are associated with a specific CMAS or CICSplex. For example, you could create a filter like this:

```
PLEXNAME=EYUPLX01.
```

and specify that filter on the LISTEN command to be notified only of events generated by CICSplex EYUPLX01.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option listens for only those events that do not meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTIFICATION*(data-ref)*
Names a variable to receive the fullword token that CICSPlex SM assigns to this notification request.

This identifying token must be specified on the CANCEL command when you want to cancel the notification request.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOKEN*(data-value)*
Defines a 1- to 4-character token that you choose to correlate this LISTEN request with the result of a subsequent RECEIVE command. This token is for use by your program; CICSPlex SM makes no use of the value. The token is returned by the RECEIVE command when an event of the specified type occurs.

Conditions

The following is a list of the RESPONSE values that can be returned by the LISTEN command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INCOMPATIBLE
An incompatible condition occurred for the following reason:

INVALIDEVT
The specified event is not compatible with the filter specified on the FILTER or NOTFILTER option.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- EVENT
- FILTER
- NOTFILTER
- NOTIFICATION
- THREAD
- TOKEN.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread was trying to connect is not available for API processing.

PLEXMGR
The CMAS to which the processing thread is currently connected does not participate in managing the specified CICSplex and no other CMAS is available that does manage the CICSplex.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

VERSIONINVL
A version conflict occurred for one of the following reasons:

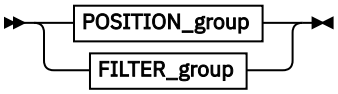
NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

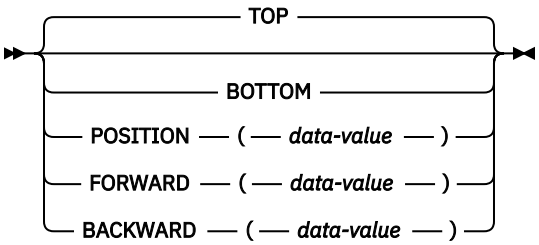
LOCATE

Position the record pointer within a result set.

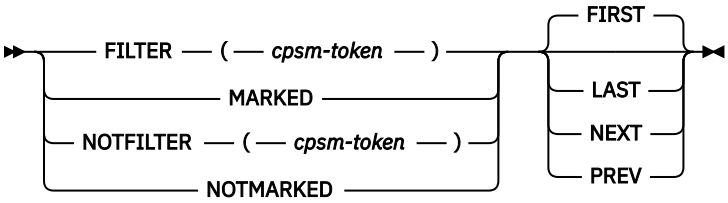
pointer_group

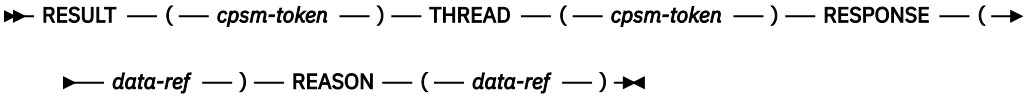


POSITION_group



FILTER_group





Description

This command positions the record pointer within a result set.

- API commands that manipulate records or update the data in a result set affect the position of the record pointer:
  - After a GET command, the pointer is positioned to the top of the result set.
  - After a FETCH command, the pointer is positioned to the next record in the result set (that is, the record following the last record fetched in whichever direction the pointer was moving, forward or backward). If no records were fetched (because no records matched the specified criteria), the pointer is positioned to the top or bottom of the result set, depending on which direction it was moving.

  After issuing any other command that manipulates records or updates data, the position of the record pointer depends on a combination of factors, including the options that you specified on the command. To be certain of the pointer's location, you should use the LOCATE command to explicitly position it within the result set.
- The LOCATE command skips over any deleted records in the result set. If you try to position the record pointer to a deleted record, you receive a RESPONSE value of NODATA.

Related commands

COPY, DELETE, FETCH, GETDEF, GROUP, MARK, ORDER, PERFORM OBJECT, PERFORM SET, REFRESH, SET, SPECIFY FILTER, UNMARK

Options

BACKWARD*(data-value)*
Moves the record pointer backward by the specified number of resource table records.

If the pointer reaches the top of the result set, it remains positioned on the first resource table record. The pointer does not continue moving backward to the bottom of the result set.

BOTTOM
Moves the record pointer to the the last resource table record in the result set.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation.

The FILTER option positions the record pointer to a resource table record that meets the specified filter criteria. The FIRST, LAST, NEXT, or PREV option determines where in the result set the search begins and in what direction it continues.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

FIRST
Begins a search based upon filter or marking criteria with the first resource table record in the result set. The search continues in a forward direction through the result set until a match is found.

FORWARD*(data-value)*
Moves the record pointer forward by the specified number of resource table records.

If the pointer reaches the bottom of the result set, it remains positioned on the last resource table record. The pointer does not continue moving forward to the top of the result set.

LAST
Begins a search based upon filter or marking criteria with the last resource table record in the result set. The search continues in a backward direction through the result set until a match is found.

MARKED
Positions the record pointer to a resource table record that is marked. The FIRST, LAST, NEXT, or PREV option determines where in the result set the search begins and in what direction it continues.

You can mark resource table records by using the MARK and UNMARK commands.

NEXT
Begins a search based upon filter or marking criteria with the current resource table record in the result set. The search continues in a forward direction through the result set until a match is found.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation.

The NOTFILTER option positions the record pointer to a resource table record that does not meet the specified filter criteria. The FIRST, LAST, NEXT, or PREV option determines where in the result set the search begins and in what direction it continues.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Positions the record pointer to a resource table record that is not marked. The FIRST, LAST, NEXT, or PREV option determines where in the result set the search begins and in what direction it continues.

You can mark resource table records by using the MARK and UNMARK commands.

POSITION*(data-value)*
Moves the record pointer to the nth resource table record in the result set.

This value must be a number that identifies the record's relative position in the result set. The first record in a result set is identified by the number 1.

For example, to move the record pointer to the fifth resource table record in a result set, you would specify `POSITION(5)`.

PREV
Begins a search based upon filter or marking criteria with the previous resource table record in the result set. The search continues in a backward direction through the result set until a match is found.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- GROUP
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOP
Moves the record pointer to the first resource table record in the result set.

Conditions

The following is a list of the RESPONSE values that can be returned by the LOCATE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria, for one of the following reasons:

BACKWARD
There are no more records that satisfy the search criteria in the backward direction.

FORWARD
There are no more records that satisfy the search criteria in the forward direction.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

SOLRESOURCE
A required resource that is locally owned (that is, owned by the address space where the processing thread is running) is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- BACKWARD
- FILTER
- FORWARD
- NOTFILTER
- POSITION
- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread was trying to connect is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

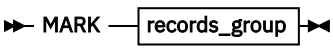
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

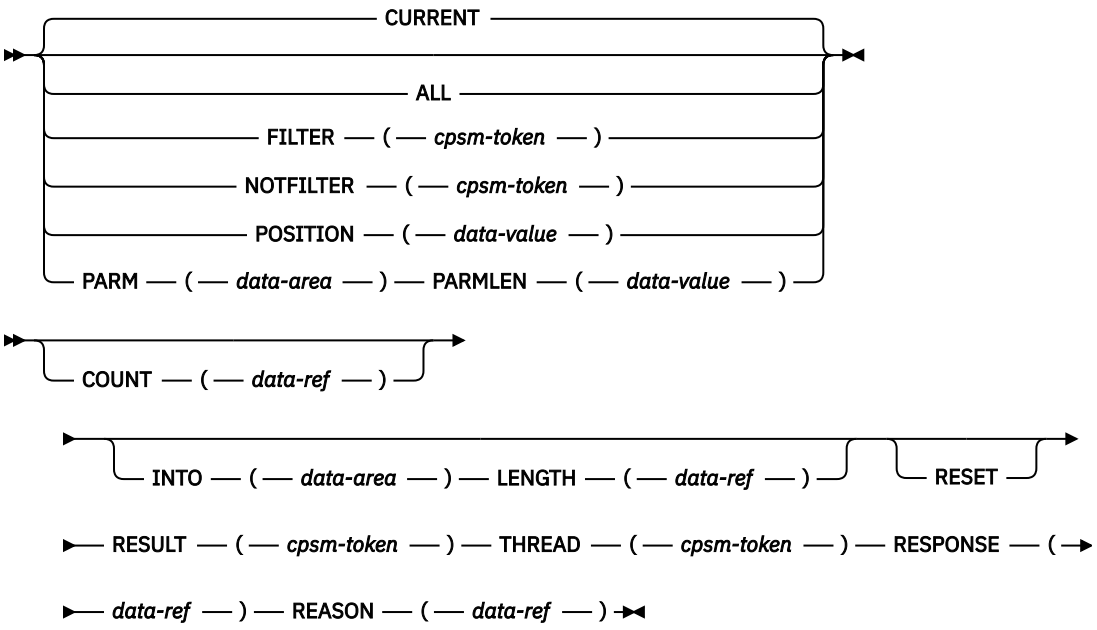
NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

MARK

Mark selected resource table records in a result set.



records_group



Description

This command marks selected resource table records in a result set.

- The MARK command always begins processing with the last record that was fetched, rather than the next one in the result set.
- Any resource table records that you marked in the result set previously remain marked unless you use the RESET option.

Related commands

COPY, DELETE, EXPAND, FETCH, GROUP, LOCATE, PERFORM SET, REFRESH, SET, SPECIFY FILTER, UNMARK

Options

ALL
Marks all the resource table records in the result set. When you specify ALL, the RESET option is ignored.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records that could not be marked.

CURRENT
Marks only the current resource table record.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option marks only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

INTO*(data-area)*
Identifies a buffer to receive a list of resource table records that could not be marked.

This buffer must be long enough to hold the maximum number of record numbers that could result from your MARK request (if none of them can be marked). Record numbers are listed individually (not by range) in the INTO buffer and are separated by commas.

Note: If you receive a RESPONSE value of WARNING AREATOOSMALL (because the buffer was not long enough), the data returned in this buffer represents a partial list of the records that could not be marked.

LENGTH*(data-ref)*
A fullword value that specifies the length of the INTO buffer.

The value that CICSPlex SM returns in this field depends on the RESPONSE value for the MARK command:

OK
The actual length of the data returned in the INTO buffer.

WARNING AREATOOSMALL
The buffer length that would be required to hold a complete list of records that could not be marked.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option marks only those resource table records that do not meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

PARM*(data-area)*
Identifies a buffer containing the parameter expression that lists the resource table records to be marked.

The parameter expression for the MARK command is a character string of record numbers. For example:

```
PARM('1,3,6:9,24.')
```

To specify individual records, separate the record numbers with a comma. To specify a range of records, separate the low and high record numbers with a colon. The whole parameter expression must end with a period.

For details on how to use a parameter expression with the MARK command, see Developing CICSPlex SM applications.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

POSITION*(data-value)*
Marks the nth resource table record in the result set.

This value must be a number that identifies the record's relative position in the result set. The first record in a result set is identified by the number 1.

For example, to mark the fifth resource table record in a result set, you would specify `POSITION(5)`.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESET
Removes any marks previously placed on resource table records in the result set and marks only those records you identify in the current MARK request.

If you do not use the RESET option, any records that you marked previously remain marked. That is, the records identified in the current MARK request are marked in addition to any previously marked records.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- GROUP
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the MARK command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria.

WARNING
The command completed processing with a warning, for one of the following reasons:

AREATOOSMALL
You specified the INTO and LENGTH options, but the buffer was not long enough to hold the string of records that could not be marked.

DATAERROR
One or more of the records specified in the PARM buffer could not be found to be marked. If you specified the COUNT option, the number of records that could not be marked is returned. If you specified the INTO and LENGTH options, a list of the records is returned in the buffer.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

SOLRESOURCE
A required resource that is locally owned (that is, owned by the address space where the processing thread is running) is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- COUNT
- FILTER
- INTO
- LENGTH
- NOTFILTER
- PARM
- PARMLEN
- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

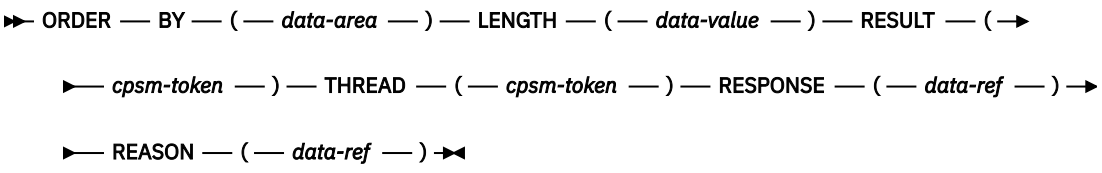
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

ORDER

Sort the resource table records in a result set.



Description

This command sorts the resource table records in a result set into a user-specified order.

- By default, records are sorted by the key attributes for the resource table.
- The sort order you specify for a result set remains in effect until you issue another ORDER command.
- If the result set contains deleted records, those records are included in the sorting process. They are sorted by the same attributes as other records and their position in the newly ordered result set may be difficult to determine. To prevent this happening, issue the REFRESH command before issuing ORDER; REFRESH removes any deleted records from the result set.

Related commands

COPY, GET, GETDEF, GROUP, LOCATE, PERFORM OBJECT

Options

BY*(data-area)*
Identifies a buffer containing the order expression to be used for this operation.

An order expression is a list of attributes to be used in sorting the resource table records. For example:

```
CICSSYS,TRANID.
```

where the attribute names are separated by commas or blank spaces and the whole expression ends with a period.

In this example, the resource table records are sorted using CICS system name as the primary sort key and transaction ID as the secondary key. The default sort order is ascending. To sort attribute values in descending order, add /D to the end of the attribute name.

For more information on using order expressions with the ORDER command, see Developing CICSPlex SM applications.

LENGTH*(data-value)*
A fullword value that specifies the length of the BY buffer.

Note: The buffer length you specify should not include any data other than an order expression.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- GROUP
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the ORDER command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- BY
- LENGTH
- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

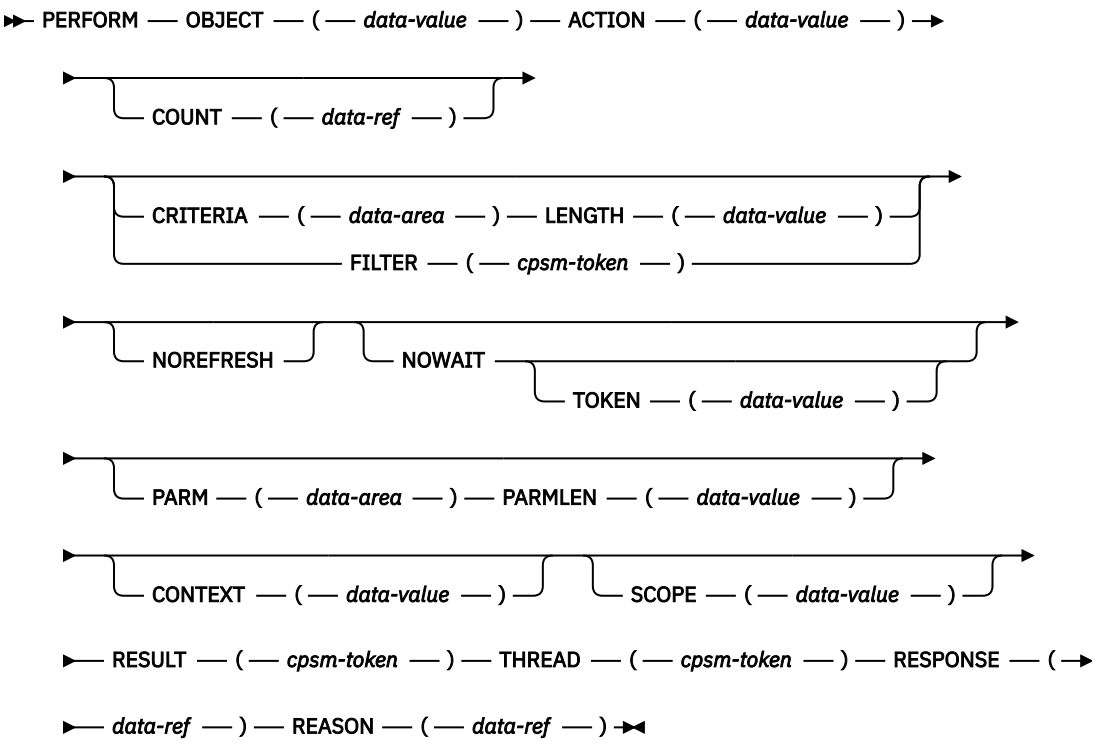
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

PERFORM OBJECT

Perform an action on one or more resources.



Description

This command performs an action on one or more resources.

- The resources to be acted upon by **PERFORM OBJECT** do not have to exist as records in a result set; a result set is implicitly created by this process.
- If the context and scope in effect when you issue a **PERFORM OBJECT** command include CICS systems that do not support the requested action, the request is ignored for those CICS systems.
- The **PERFORM OBJECT** command contains two phases; the first is to build the result set, and the second is to take the requested action against the records in the result set. If an error occurs during the building of the result set, but the result set is not empty, the requested action will still be attempted on the records that are present in the result set.

Related commands

DISCARD, GET, LOCATE, PERFORM SET, QUERY, SET, SPECIFY FILTER

Options

ACTION*(data-value)*
Identifies the action to be performed. This value must be the 1- to 12-character name of a valid action for the resource table.

For a description of the actions that are valid for a given resource table, see CICSPlex SM resource tables in Reference.

CONTEXT*(data-value)*
Identifies the context for this command. The context must be the 1- to 8-character name of a CMAS or CICSplex.

If you do not specify the CONTEXT option, the default context for the thread is assumed.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records in the target result set after this operation is complete.

CRITERIA*(data-area)*
Identifies a buffer containing the filter expression to be used for this operation. The CRITERIA option retrieves only those resource table records that meet the specified filter criteria.

For details on how to form a filter expression, see Developing CICSPlex SM applications.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option retrieves only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

LENGTH*(data-value)*
A fullword value that specifies the length of the CRITERIA buffer.

Note: The buffer length you specify should not include any data other than a filter expression.

NOREFRESH
Specifies that the resource table records in the result set created by PERFORM OBJECT should not be refreshed. The records reflect the status of the resources when the result set was created.

If you do not specify the NOREFRESH option, the resource table records are refreshed to reflect the resource status after this operation is complete.

NOWAIT
Returns control to your program as soon as the PERFORM OBJECT command has been accepted, which allows the command to be processed asynchronously.

If you specify the NOWAIT option, you must use a subsequent RECEIVE command to test for the completion of this request. The results of an asynchronous request are returned as ASYNCREQ resource table records. For a complete description of asynchronous processing, see Developing CICSPlex SM applications.

Note: If you specify the TOKEN option, the NOWAIT option is assumed by default.

OBJECT*(data-value)*
Identifies the resource table against which the action is to be performed. This value must be the 8-character name of a valid resource table.

PARM*(data-area)*
Identifies a buffer containing the parameter expression to be used in performing the action.

For details on how to use a parameter expression with the PERFORM OBJECT command, see Developing CICSPlex SM applications. For a description of the parameters that are required for a given resource table action, see CICSPlex SM resource tables in Reference.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GROUP

- PERFORM OBJECT.

CICSPlex SM replaces the contents of the existing result set with the resource table records requested by this **PERFORM OBJECT** command.

If this field is:

- Set to binary zero (in COBOL, C, PL/I or Assembler)
- An uninitialized variable (in REXX).

CICSPlex SM creates a new result set and returns its identifying token in the same field.

SCOPE*(data-value)*
Identifies the scope for this command.

To use the SCOPE option, the current context (as set by this command or a previous CONNECT or QUALIFY command) must be a CICSplex. The scope can be:

- The 1- to 8-character name of the CICSplex itself
- A CICS system or CICS system group within the CICSplex
- A logical scope, as defined in a CICSPlex SM resource description (RESDESC).

If OBJECT identifies a CICS definitional resource and the PARM option includes the **CSDGROUP** parameter, a valid scope can be specified. The scope can be:

- A CICS system in the CICSplex.

If the current context is a CMAS or the OBJECT option identifies any other type of resource table, this option has no meaning and is ignored.

If you do not specify the SCOPE option, the default scope for the thread is assumed. If the current context is a CICSplex and no default scope has been set on a CONNECT or QUALIFY command, you receive an INVALIDPARM response for the SCOPE option.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOKEN*(data-value)*
Defines a 1- to 4-character token that you choose to correlate an asynchronous **PERFORM OBJECT** request with the result of a subsequent RECEIVE command. This token is for use by your program; CICSPlex SM makes no use of the value. The token is returned by the RECEIVE command when this **PERFORM OBJECT** request is complete.

Conditions

The following is a list of the RESPONSE values that can be returned by the **PERFORM OBJECT** command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

SCHEDULED
The command has been scheduled for processing.

NODATA
No records were found that matched the specified search criteria.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
One of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs or MASs to which the request was directed did not respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDATA
Invalid data was detected. The parameter that contains invalid data is returned as the reason value:

PARM
An attribute value listed in the PARM buffer is not valid for the specified attribute.

CRITERIA
An attribute value listed in the CRITERIA buffer is not valid for the specified attribute.

INVALIDCMD
The command is invalid for one of the following reasons:

FILTER
The filter expression passed on the operation is too large or complex.

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- ACTION
- CONTEXT
- CRITERIA
- FILTER
- LENGTH
- OBJECT
- PARM
- PARMLEN
- RESULT
- SCOPE
- THREAD
- TOKEN.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

MAINTPOINT
The maintenance point for the current context is not available.

PLEXMGR
The CMAS to which the processing thread is currently connected does not participate in managing the specified CICSplex and no other CMAS is available that does manage the CICSplex.

SCOPE
Either none of the MASs in the specified scope are available or none of them support the requested action.

WORKLOAD
The workload identified on the API request is not available on the local CMAS.

NOTFOUND
A not found condition occurred for one of the following reasons:

ACTION
The action specified on the ACTION option was not found for the specified resource table.

ATTRIBUTE
An attribute specified in the CRITERIA or PARM buffer was not found for the specified resource table.

NOTPERMIT
A not permitted condition occurred for the following reason:

USRID
The user ID associated with the processing thread does not have the required security authorization.

TABLEERROR
A resource table record is invalid for the following reason:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if the resource table is missing required fields, contains one or more conflicting fields, or is a duplicate. For BAS this error can also occur if you do not have the required security authorization. Use the FEEDBACK command to retrieve additional data about this error.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

VERSIONINVL
A version conflict occurred for one of the following reasons:

ACTION
The specified action is not supported for the version used with the CONNECT command.

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

WARNING
The PERFORM OBJECT command may have only partially completed for one of the following reasons:

RESULT
During the building of the result set to be used on the command, a non-OK response was received. However some result set records were available and the requested action was successfully performed against them. Use the FEEDBACK command without the RESULT option to obtain information about the non-OK response.

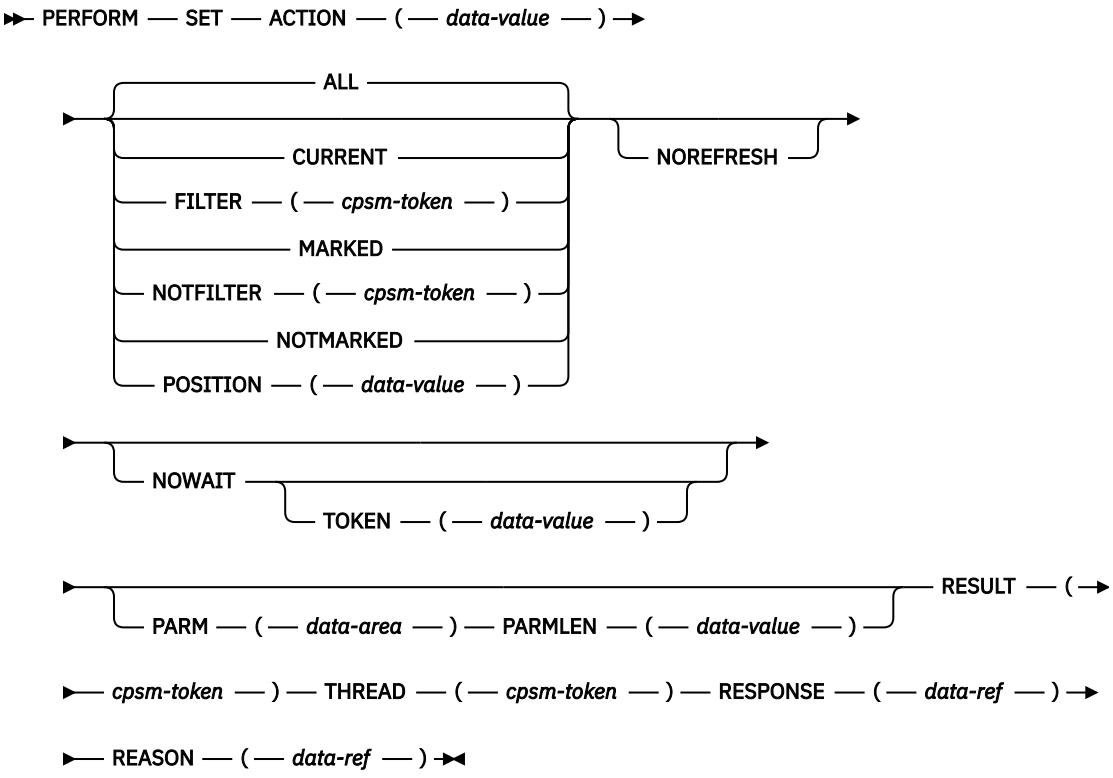
ACTION
During the building of the result set to be used on the command, a non-OK response was received. However some result set records were available and the requested action was attempted. The action specified did not complete successfully on, at least one result set record due to a TABLEERROR or DATAERROR CICSPlex SM response or reason.

Use the FEEDBACK command without the RESULT option to obtain information about the error that occurred during the building of the result set. Use the FEEDBACK command with the RESULT option to obtain information about records that caused the TABLEERROR or DATAERROR response or reason.

WORKLOAD
The workload identified on the API request is not available on the local CMAS.

PERFORM SET

Perform an action on one or more resources.



Description

This command performs an action on one or more resources as represented by resource table records in an existing result set. If the context and scope in effect when you issue a PERFORM SET command include CICS systems that do not support the requested action, the request is ignored for those CICS systems.

Related commands

LOCATE, MARK, PERFORM OBJECT, SET, SPECIFY FILTER

Options

ACTION*(data-value)*
Identifies the action to be performed. This value must be the 1- to 12-character name of a valid action for the resource table.

For a description of the actions that are valid for a given resource table, see CICSPlex SM resource tables in Reference.

ALL
Performs the specified action against all the resource table records in the result set.

CURRENT
Performs the specified action against only the current resource table record.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option performs the action against only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

MARKED
Performs the specified action against only those resource table records that are marked in the result set. You can mark resource table records by using the MARK and UNMARK commands.

NOREFRESH
Specifies that the resource table records in the source result set should not be refreshed. The records reflect the status of the resources before the PERFORM SET command was processed.

If you do not specify the NOREFRESH option, the resource table records are refreshed to reflect the resource status after this operation is complete.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option performs the action against only those resource table records that do not meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Performs the specified action against only those resource table records that are not marked in the result set. You can mark resource table records by using the MARK and UNMARK commands.

NOWAIT
Returns control to your program as soon as the PERFORM SET command has been accepted, which allows the command to be processed asynchronously.

If you specify the NOWAIT option, you must use a subsequent RECEIVE command to test for the completion of this request. The results of an asynchronous request are returned as ASYNCREQ resource table records. For a complete description of asynchronous processing, see Developing CICSPlex SM applications.

Note: If you specify the TOKEN option, the NOWAIT option is assumed by default.

PARM*(data-area)*
Identifies a buffer containing the parameter expression to be used in performing the action.

For details on how to use a parameter expression with the PERFORM SET command, see Developing CICSPlex SM applications. For a description of the parameters that are required for a given resource table action, see CICSPlex SM resource tables in Reference.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

POSITION*(data-value)*
Performs the specified action against the nth resource table record in the result set.

This value must be a number that identifies the record's relative position in the result set. The first record in a result set is identified by the number 1.

For example, to perform the specified action on the fifth resource table record in a result set, you would specify `POSITION(5)`.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GROUP
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOKEN*(data-value)*
Defines a 1- to 4-character token that you choose to correlate an asynchronous PERFORM SET request with the result of a subsequent RECEIVE command. This token is for use by your program; CICSPlex SM makes no use of the value. The token is returned by the RECEIVE command when this PERFORM SET request is complete.

Conditions

The following is a list of the RESPONSE values that can be returned by the PERFORM SET command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

SCHEDULED
The command has been scheduled for processing.

NODATA
No records were found that matched the specified search criteria. If the ALL option was specified, the following reason may be returned:

FORWARD
There are no more records that satisfy the search criteria in the forward direction.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
One of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs or MASs to which the request was directed did not respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDATA
Invalid data was detected. The parameter that contains invalid data is returned as the reason value:

PARM
An attribute value listed in the PARM buffer is not valid for the specified attribute.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- ACTION
- FILTER
- NOTFILTER
- PARM
- PARMLEN
- POSITION
- RESULT
- THREAD
- TOKEN.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

MAINTPOINT
The maintenance point for the current context is not available.

PLEXMGR
The CMAS to which the processing thread is currently connected does not participate in managing the specified CICSplex and no other CMAS is available that does manage the CICSplex.

SCOPE
At least one of the MASs in the specified scope is unavailable

WORKLOAD
The workload identified on the API request is not available on the local CMAS.

NOTFOUND
A not found condition occurred for one of the following reasons:

ACTION
The action specified on the ACTION option was not found for the specified resource table.

ATTRIBUTE
An attribute specified in the CRITERIA or PARM buffer was not found for the specified resource table.

NOTPERMIT
A not permitted condition occurred for the following reason:

USRID
The user ID associated with the processing thread does not have the required security authorization.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for the following reason:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if the resource table is missing required fields, contains one or more conflicting fields, or is a duplicate. For BAS this error can also occur if you do not have the required security authorization. Use the FEEDBACK command to retrieve additional data about this error.

VERSIONINVL
A version conflict occurred for one of the following reasons:

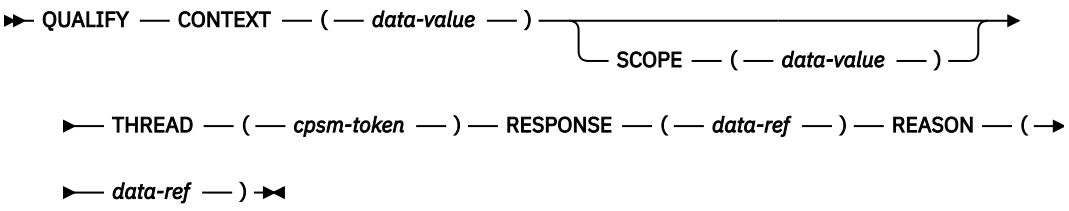
ACTION
The specified action is not supported for the version used with the CONNECT command.

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

QUALIFY

Define the CICSPlex SM context and scope.



Related commands

CONNECT

Description

This command defines the CICSPlex SM context and scope for subsequent commands issued by an API processing thread.

Options

CONTEXT*(data-value)*
Identifies the context for subsequent commands issued against this thread. The context must be the 1- to 8-character name of a CMAS or CICSplex.

The specified context remains in effect for the thread until you override it or change it on a subsequent command.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

SCOPE*(data-value)*
Identifies the scope for subsequent commands issued against this thread.

The SCOPE option qualifies the CONTEXT option. When the context is a CICSplex, the scope can be:

- The 1- to 8-character name of the CICSplex itself
- A CICS system or CICS system group within the CICSplex
- A logical scope, as defined in a CICSPlex SM resource description (RESDESC).

When the context is a CMAS, this option has no meaning and is ignored.

The specified scope remains in effect for the thread unless you override it for a specific command or change it by issuing another QUALIFY command. If you do not specify the SCOPE option, no scope value is assumed (that is, the default scope established for the thread by the CONNECT command is not retained).

Note: Certain API commands require a valid scope when the context is a CICSplex. If you do not specify a scope on the QUALIFY command, then you must specify the SCOPE option when you issue any of these commands for a resource table that represents a CICS resource:

- GET
- PERFORM OBJECT
- PERFORM SET
- REFRESH
- SET.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the QUALIFY command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- SCOPE
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

PLEXMGR
The CMAS to which the processing thread is currently connected does not participate in managing the specified CICSplex and no other CMAS is available that does manage the CICSplex.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

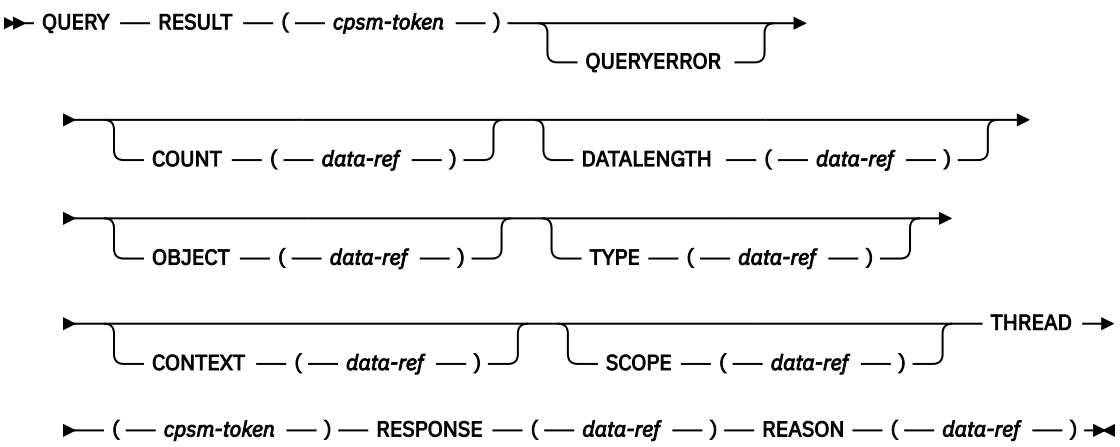
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

QUERY

Retrieve information about a result set and the resource table records it contains.



Description

This command retrieves information about a result set and the resource table records it contains.

- You can use the QUERY command to determine:
  - The context and scope of the result set
  - The type of resource table records the result set contains
  - Whether the records are from the CICSPlex SM resource table or a user-defined view of that table
  - The number of resource table records in the result set
  - The length of the resource table records
- For programs written in REXX, issuing the QUERY command is the only way to determine the length of a given resource table record.

Related commands

COPY, GET, GETDEF, GROUP, PERFORM OBJECT

Options

CONTEXT*(data-ref)*
Names a variable to receive the context associated with the result set.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records in the result set.

DATALENGTH*(data-ref)*
Names a variable to receive the length of the resource table records in the result set.

OBJECT*(data-ref)*
Names a variable to receive the name of the resource table currently associated with the result set.

Note: If QUERYERROR is specified, the OBJECT returned is MASQRYER, not the object or view contained in the result set.

QUERYERROR
Indicates that this request is to return information on MASQRYER resources generated by the last GET, PERFORM, or SET command to act on the result set.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- GROUP
- PERFORM OBJECT.

SCOPE*(data-ref)*
Names a variable to receive the scope associated with the result set. This value may be blank for result sets containing CMAS type resources.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TYPE*(data-ref)*
Names a variable to receive a 1-character value that indicates what type of records are in the result set:

T
Resource tables supplied by CICSPlex SM.

V
Views of a resource table created by a SPECIFY VIEW command issued previously on this processing thread.

Conditions

The following is a list of the RESPONSE values that can be returned by the QUERY command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- DATALENGTH
- OBJECT
- RESULT
- THREAD
- TYPE.

Check the command description for valid parameter syntax.

NODATA
The command requested information about MASQRYER resources generated by the last command to process the result set, but the last command completed successfully, and there are no MASQRYER resources. If COUNT was requested a value of zero is returned.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

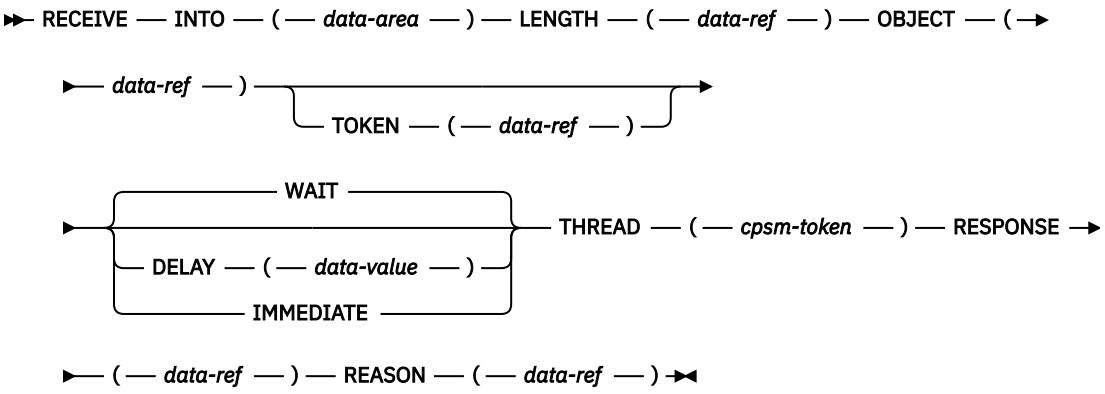
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

RECEIVE

Receive the output from completed asynchronous requests.



Description

This command receives the output from completed asynchronous requests associated with the processing thread.

- Asynchronous output can result if you previously issued either a LISTEN command or one of these commands with the NOWAIT option:
  - GET
  - PERFORM OBJECT
  - PERFORM SET
  - REFRESH
  - SET.
- To determine if there is any asynchronous output to be received, issue the ADDRESS command and check the SENTINEL value before you issue the RECEIVE command.
- An API processing thread can have a maximum of 256 completed asynchronous requests outstanding at one time. If you do not issue the RECEIVE command at regular intervals and your processing thread reaches its maximum of 256, asynchronous requests are discarded and are not processed. For a complete description of asynchronous processing, see Developing CICSPlex SM applications.

Related commands

ADDRESS, GET, LISTEN, PERFORM OBJECT, PERFORM SET, REFRESH, SET

Options

DELAY*(data-value)*
Specifies the number of seconds that processing will wait if no output is available when the RECEIVE command is issued. At the end of the specified number of seconds, control returns to the processing thread, whether or not any output becomes available. If output becomes available during the delay period, control returns to the processing thread. If output is immediately available, there is no delay; control returns immediately to the processing thread.

DELAY must specify a non-zero value. If you want to make sure that your program never enters a wait, use the IMMEDIATE option instead of DELAY.

IMMEDIATE
Returns control to the processing thread immediately, whether or not any output is available.

INTO*(data-area)*
Identifies a buffer to receive asynchronous output, if any is available for this thread. This buffer must be long enough to hold all the output being received.

The output returned can be:

- A resource table record representing an event named in a previous LISTEN command
- An ASYNCREQ resource table record representing completion of an asynchronous GET, PERFORM, REFRESH, or SET request.

LENGTH*(data-ref)*
A fullword value that specifies the length of the INTO buffer.

After the operation is complete, this field is set to the actual length of the data returned in the INTO buffer. If the operation cannot complete because the buffer is not long enough, this field is set to the length that is required.

OBJECT*(data-ref)*
Names a variable to receive a resource table name, if output is available for this thread.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

WAIT
Waits until asynchronous output becomes available before returning control to the processing thread.

Note: The WAIT option waits indefinitely for asynchronous output. Be sure to verify that there are completed asynchronous requests outstanding by issuing the ADDRESS command before you issue RECEIVE.

TOKEN*(data-ref)*
Names a variable to receive the user-defined token associated with the asynchronous output. This value is the token you defined on the GET, LISTEN, PERFORM, REFRESH or SET command that produced the output.

Conditions

The following is a list of the RESPONSE values that can be returned by the RECEIVE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
There was no data to receive.

WARNING
The command completed processing with a warning, for the following reason:

AREATOOSMALL
The INTO buffer is not long enough to hold the number of records requested and available.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- DELAY
- INTO
- LENGTH
- OBJECT
- THREAD
- TOKEN.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

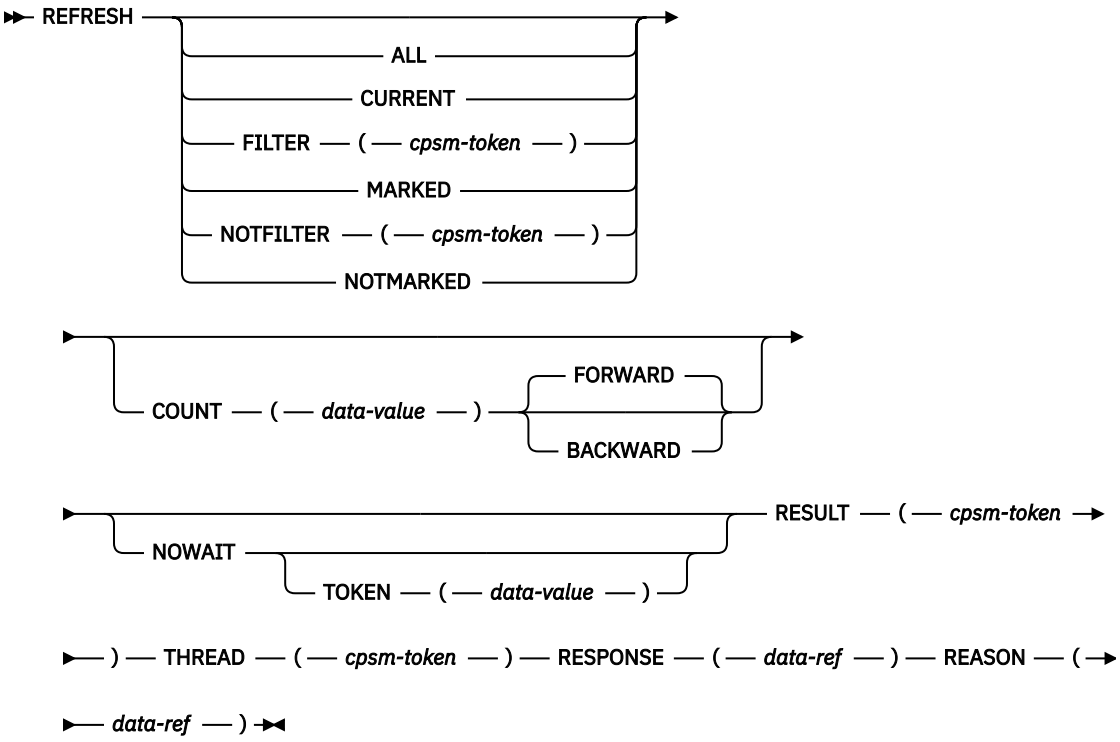
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

REFRESH

Refresh the data for resource table records.



Description

- This command refreshes the data for some or all of the resource table records in a result set.
- For the MAS resource table, REFRESH provides data only if the MAS was active when the result set was last built.

Related commands

COPY, GET, LOCATE, MARK, PERFORM OBJECT, SPECIFY FILTER

Options

ALL
Refreshes all the resource table records in the result set. When you specify ALL:

- The COUNT option is ignored.
- Any records that have been deleted are removed from the result set. Any positions previously held by deleted records are filled in and the remaining records are renumbered. Therefore, the relative position of a given record in a result set may be different after a refresh.

BACKWARD
Refreshes the previous resource table record and continues in a backward direction through the result set refreshing as many records as the COUNT option specifies.

Note: If the record pointer is at the bottom of the result set, using BACKWARD refreshes the current record (which is the last record) and then continues on to previous records.

COUNT*(data-value)*
Specifies the number of resource table records to be refreshed. If you do not specify the COUNT option, only one record is refreshed.

If you do not specify the FORWARD or BACKWARD option, the refresh process moves in a forward direction through the result set.

CURRENT
Refreshes only the current resource table record. When you specify CURRENT, the COUNT option is ignored.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option indicates that only those resource table records that meet the specified filter criteria should be considered for refresh.

The number of records that are refreshed is determined by the COUNT option. If you do not specify the COUNT option, only the first record that meets the filter criteria is refreshed.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

FORWARD
Refreshes the current resource table record and continues in a forward direction through the result set refreshing as many records as the COUNT option specifies.

MARKED
Indicates that only those resource table records that are marked in the result set should be considered for refresh.

The number of records that are refreshed is determined by the COUNT option. If you do not specify the COUNT option, only the first record that is marked is refreshed.

You can mark resource table records by using the MARK and UNMARK commands.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option indicates that only those resource table records that do not meet the specified filter criteria should be considered for refresh.

The number of records that are refreshed is determined by the COUNT option. If you do not specify the COUNT option, only the first record that does not meet the filter criteria is refreshed.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Indicates that only those resource table records that are not marked in the result set should be considered for refresh.

The number of records that are refreshed is determined by the COUNT option. If you do not specify the COUNT option, only the first record that is not marked is refreshed.

You can mark resource table records by using the MARK and UNMARK commands.

NOWAIT
Returns control to your program as soon as the REFRESH command has been accepted, which allows the command to be processed asynchronously.

If you specify the NOWAIT option, you must use a subsequent RECEIVE command to test for the completion of this request. The results of an asynchronous request are returned as ASYNCREQ resource table records. For a complete description of asynchronous processing, see Developing CICSPlex SM applications.

Note: If you specify the TOKEN option, the NOWAIT option is assumed by default.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOKEN*(data-value)*
Defines a 1- to 4-character token that you choose to correlate an asynchronous REFRESH request with the result of a subsequent RECEIVE command. This token is for use by your program; CICSPlex SM makes no use of the value. The token is returned by the RECEIVE command when this REFRESH request is complete.

Conditions

The following is a list of the RESPONSE values that can be returned by the REFRESH command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

SCHEDULED
The command has been scheduled for processing.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
One of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs or MASs to which the request was directed did not respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- COUNT
- FILTER
- NOTFILTER
- RESULT
- THREAD
- TOKEN.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

MAINTPOINT
The maintenance point for the current context is not available.

RESOURCE
The resource table type for the result set being refreshed is not supported by this command. To refresh the result set, re-execute the GET command used to originally acquire it. Currently, result sets for the EVCSPEC, HTASK, OSGIBUND, OSGISERV, SYSPARM, and RULE resource tables are not refreshable.

SCOPE
One or more of the MASs in the specified scope is unavailable. A MASQRYER resource table record is produced to identify any MAS that did not respond to the request.

NOTPERMIT
A not permitted condition occurred for the following reason:

USRID
The user ID associated with the processing thread does not have the required security authorization.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for the following reason:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if the resource table is missing required attributes, contains one or more conflicting attributes, or does not exist. Use the FEEDBACK command to retrieve additional data about this error.

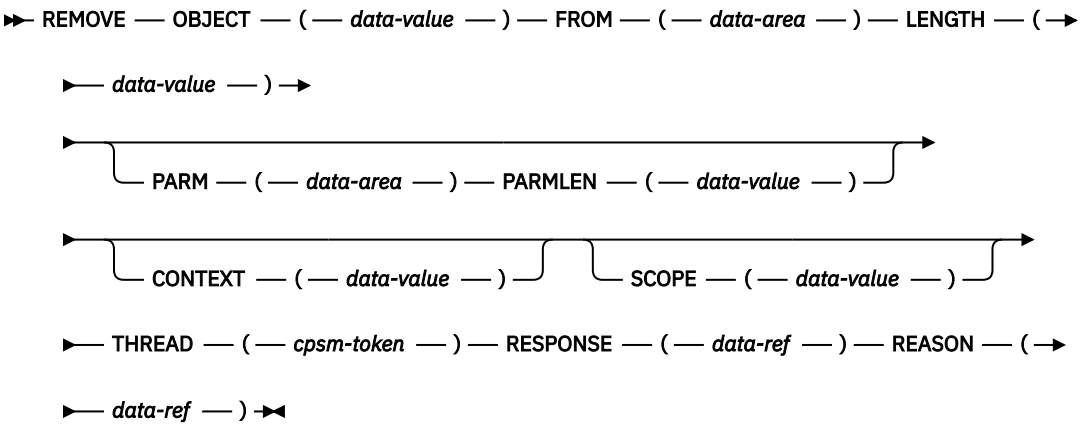
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

REMOVE

Remove a CICSPlex SM or CICS definition from the data repository.



Description

This commands removes a CICSPlex SM or CICS definition from the data repository. For definitions that have a CICSplex as their context (such as workload management or real-time analysis definitions), the definition is also removed from the data repositories of all CMASs involved in managing the CICSplex.

Related commands

CREATE, UPDATE

Options

CONTEXT*(data-value)*
Identifies the context for this command. The context must be the 1- to 8-character name of a CMAS or CICSplex.

If you do not specify the CONTEXT option, the default context for the thread is assumed.

FROM*(data-area)*
Identifies a buffer containing a resource table record that represents the definition to be removed. The record must include all of the attributes for the resource table specified on the OBJECT option.

LENGTH*(data-value)*
A fullword value that specifies the length of the FROM buffer.

OBJECT*(data-value)*
Identifies the resource table that represents the definition being removed. This value must be the 1- to 8-character name of a valid CICSPlex SM Definition or CICS Definition resource table. For a list of the CICSPlex SM resource tables by type, see Developing CICSPlex SM applications.

PARM*(data-area)*
Identifies a buffer containing the parameter expression to be used in removing the definition.

For details on how to use a parameter expression with the REMOVE command, see Developing CICSPlex SM applications. For a description of the parameters that are valid for a given resource table, see the CICSPlex SM resource tables in Reference.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

SCOPE*(data-value)*
Identifies the scope for this command.

If the current context (as set by this command or a previous CONNECT or QUALIFY command) is a CICSplex and the OBJECT option identifies a CICS definitional resource and the PARM option includes the CSD parameter, a valid scope can be specified.

The scope can be a CICS system within the CICSplex. If the current context is a CMAS or the OBJECT option identifies any other type of resource table, or the CSD parameter is not specified on a CICS definitional resource, this option has no meaning and is ignored.

If SCOPE applies to the command and you do not specify the SCOPE option, the default scope for the thread is assumed. If the current context is a CICSplex and no default scope has been set on a CONNECT or QUALIFY command, you receive an INVALIDPARM response for the SCOPE option.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the REMOVE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
For CSD requests only, one of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs to which the request was directed did not respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- FROM
- LENGTH
- OBJECT
- PARM
- PARMLEN
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

CSDAPI
Support for the CICSPlex SM API to access the CICS CSD is not available.

MAINTPOINT
The maintenance point for the current context is not available.

NOTPERMIT
A not permitted condition occurred for the following reason:

USRID
The user ID associated with the processing thread does not have the required security authorization.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for one of the following reasons:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if the resource table is missing required attributes, contains one or more conflicting attributes, or does not exist. Use the FEEDBACK command to retrieve additional data about this error.

INVALIDATTR
One of the resource table attributes is invalid.

INVALIDVER
The specified version of the resource table is not supported by CICSPlex SM.

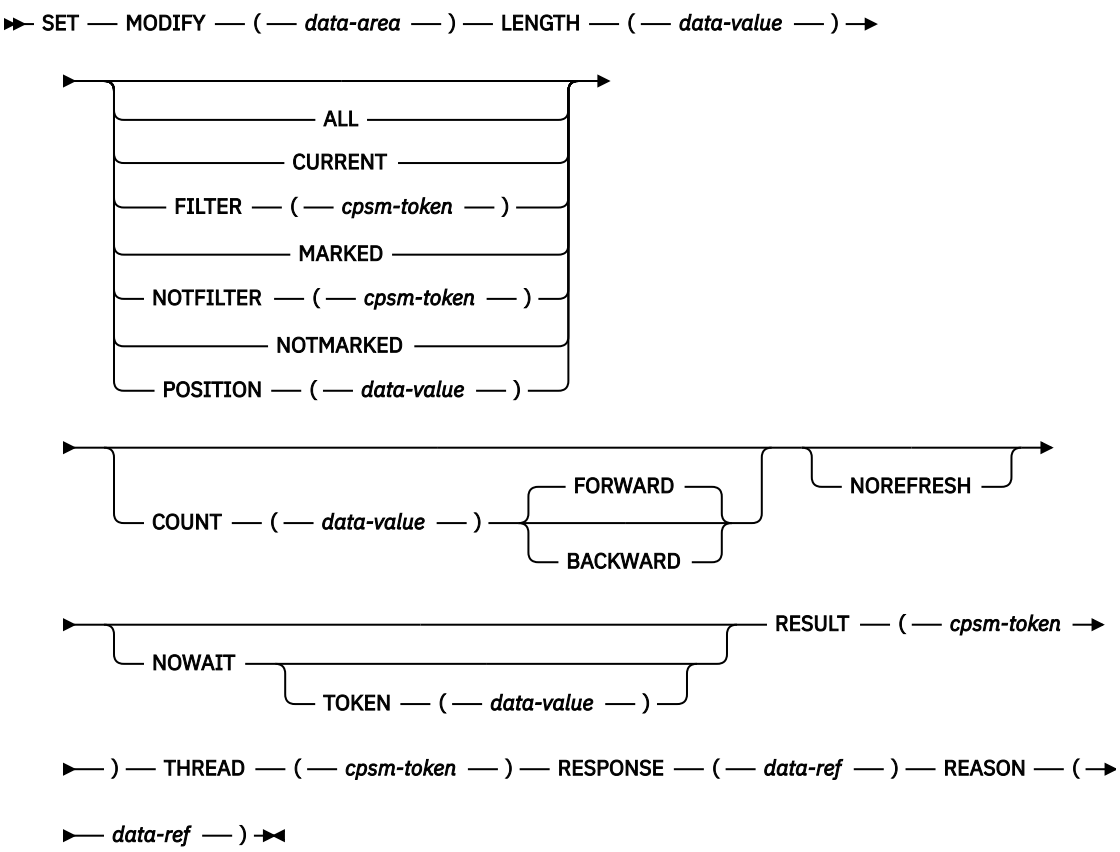
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

SET

Modify the attributes of one or more resources.



Description

This command modifies the attributes of one or more resources as represented by resource table records in an existing result set.

- The SET command is valid only for CICS Resource and some CICSPlex SM resource tables.
- If the context and scope in effect when you issue a SET command include CICS systems that do not support the requested modification, the request is ignored for those CICS systems.

Related commands

COPY, GET, GROUP, LOCATE, MARK, PERFORM OBJECT, PERFORM SET, SPECIFY FILTER

Options

ALL
Modifies all the resource table records in the result set. When you specify ALL, the COUNT option is ignored.

BACKWARD
Modifies the previous resource table record and continues in a backward direction through the result set modifying as many records as the COUNT option specifies.

Note: If the record pointer is at the bottom of the result set, using BACKWARD modifies the current record (which is the last record) and then continues on to previous records.

COUNT*(data-value)*
Specifies the number of resource table records to be modified. If you do not specify the COUNT option, only one record is refreshed.

If you do not specify the FORWARD or BACKWARD option, the modification process moves in a forward direction through the result set.

CURRENT
Modifies only the current resource table record. When you specify CURRENT, the COUNT option is ignored.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option indicates that only those resource table records that meet the specified filter criteria should be considered for modification.

The number of records that are modified is determined by the COUNT option. If you do not specify the COUNT option, only the first record that meets the filter criteria is modified.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

FORWARD
Modifies the current resource table record and continues in a forward direction through the result set modifying as many records as the COUNT option specifies.

LENGTH*(data-value)*
A fullword value that specifies the length of the MODIFY buffer.

Note: The buffer length you specify should not include any data other than a modification expression.

MARKED
Indicates that only those resource table records that are marked in the result set should be considered for modification.

The number of records that are modified is determined by the COUNT option. If you do not specify the COUNT option, only the first record that is marked is modified.

You can mark resource table records by using the MARK and UNMARK commands.

MODIFY*(data-area)*
Identifies a buffer containing the modification expression to be used in modifying the resource table records.

For details on how to form a modification expression, see Developing CICSPlex SM applications.

NOREFRESH
Specifies that the resource table records in the source result set should not be refreshed. The records reflect the status of the resources before the SET command was processed.

If you do not specify the NOREFRESH option, the resource table records are refreshed to reflect the resource status after this operation is complete.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option indicates that only those resource table records that do not meet the specified filter criteria should be considered for modification.

The number of records that are modified is determined by the COUNT option. If you do not specify the COUNT option, only the first record that does not meet the filter criteria is modified.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

NOTMARKED
Indicates that only those resource table records that are not marked in the result set should be considered for modification.

The number of records that are modified is determined by the COUNT option. If you do not specify the COUNT option, only the first record that is not marked is modified.

You can mark resource table records by using the MARK and UNMARK commands.

NOWAIT
Returns control to your program as soon as the SET command has been accepted, which allows the command to be processed asynchronously.

If you specify the NOWAIT option, you must use a subsequent RECEIVE command to test for the completion of this request. The results of an asynchronous request are returned as ASYNCREQ resource table records. For a complete description of asynchronous processing, see Developing CICSPlex SM applications.

Note: If you specify the TOKEN option, the NOWAIT option is assumed by default.

POSITION*(data-value)*
Modifies the nth resource table record in the result set. When you specify POSITION, the COUNT option is ignored.

This value must be a number that identifies the record's relative position in the result set. The first record in a result set is identified by the number 1.

For example, to modify the fifth resource table record in a result set, you would specify `POSITION(5)`.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GROUP
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOKEN*(data-value)*
Defines a 1- to 4-character token that you choose to correlate an asynchronous SET request with the result of a subsequent RECEIVE command. This token is for use by your program; CICSPlex SM makes no use of the value. The token is returned by the RECEIVE command when this SET request is complete.

Conditions

The following is a list of the RESPONSE values that can be returned by the SET command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

SCHEDULED
The command has been scheduled for processing.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
One of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs or MASs to which the request was directed did not respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDATA
An invalid data error occurred for one of the following reasons:

MODIFY
An attribute value listed in the MODIFY buffer is not valid for the specified attribute.

NOTSUPPORTED
An attribute listed in the MODIFY buffer is not modifiable.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected in either the command string or the MODIFY buffer. The parameter that is invalid is returned as the reason value:

- ATTRIBUTE
- COUNT
- FILTER
- LENGTH
- MODIFY
- NOTFILTER
- POSITION
- RESULT
- THREAD
- TOKEN.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

MAINTPOINT
The maintenance point for the current context is not available.

SCOPE
Either none of the MASs in the specified scope are available or none of them support the requested modification.

NOTFOUND
A not found condition occurred for one of the following reasons:

ACTION
An action requested in the MODIFY buffer was not found for the specified resource table.

ATTRIBUTE
An attribute specified in the MODIFY buffer was not found for the specified resource table.

NOTPERMIT
A not permitted condition occurred for the following reason:

USRID
The user ID associated with the processing thread does not have the required security authorization.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for one of the following reasons:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if the resource table is missing required attributes, contains one or more conflicting attributes, or is a duplicate. Use the FEEDBACK command to retrieve additional data about this error.

INVALIDVER
The specified version of the resource table is not supported by CICSPlex SM.

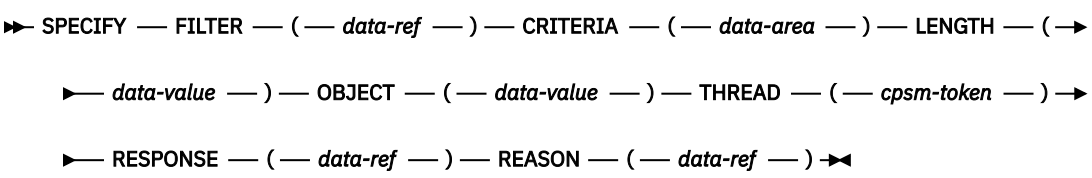
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

SPECIFY FILTER

Define an attribute or value filter and assign an identifying token to it.



Description

This command defines an attribute or value filter and assigns an identifying token to it.

- Filters are associated with the specific processing thread on which they are defined; they cannot be shared by other processing threads.
- You can define multiple filters for use by a processing thread; CICSPlex SM assigns a unique identifying token to each one.
- When a processing thread is terminated, any filters defined by it are discarded.

Related commands

COPY, DELETE, DISCARD, FETCH, GET, GROUP, LISTEN, LOCATE, MARK, PERFORM OBJECT, PERFORM SET, REFRESH, SET, UNMARK

Options

CRITERIA*(data-area)*
Identifies a buffer containing the filter expression to be used for this operation. The CRITERIA option filters only those resource table records that meet the specified criteria.

For details on how to form a filter expression, see Developing CICSPlex SM applications.

Note: You cannot specify the EYU_CICSNAME or EYU_CICSREL attributes in a filter expression.

FILTER*(data-ref)*
Names a variable to receive the token that CICSPlex SM assigns to this filter.

This identifying token must be specified on all subsequent commands that use this filter.

LENGTH*(data-value)*
A fullword value that specifies the length of the CRITERIA buffer.

Note: The buffer length you specify should not include any data other than a filter expression.

OBJECT*(data-value)*
Identifies the resource table for which a filter is being created. This value must be the 8-character name of a valid resource table.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the SPECIFY FILTER command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDATA
Invalid data was detected. The parameter that contains invalid data is returned as the reason value:

CRITERIA
An attribute value listed in the CRITERIA buffer is not valid for the specified attribute.

INVALIDCMD
The command is invalid for one of the following reasons:

FILTER
The filter expression passed on the operation is too large or complex.

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CRITERIA
- FILTER
- LENGTH
- OBJECT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

NOTFOUND
A not found condition occurred for the following reason:

ATTRIBUTE
An attribute specified in the CRITERIA buffer was not found for the specified resource table.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

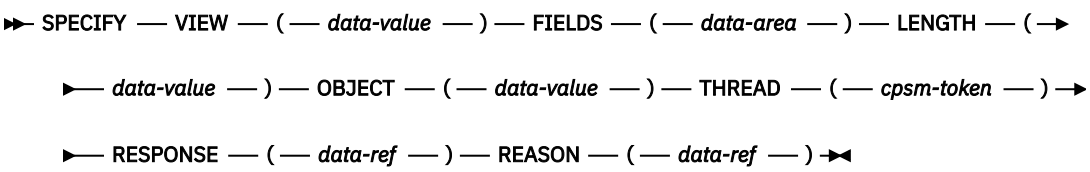
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

SPECIFY VIEW

Build a customized view of a given resource table.



Description

This command builds a customized view of a given resource table.

- Views can be built only for resource tables with a type of CICS Resource.
- Views are associated with the specific processing thread on which they are built; they cannot be shared by other processing threads.
- When a processing thread is terminated, any views built by it are deleted.
- The name you assign to a view takes precedence over any existing resource table names. You can redefine an existing resource table name to represent a customized view of that resource table.
- You are recommended to use names for customized views that are not already assigned either to other customized views or to CICSPlex SM-supplied resource tables. If you do use a name that is already assigned, you should be aware that your processing could be affected. For more details, see Developing CICSPlex SM applications.
- When you upgrade to a later version of CICSPlex SM, you should check that any new resource tables do not have the same names as any customized views. For more details, see Developing CICSPlex SM applications.

Related commands

DISCARD, GET

Options

FIELDS*(data-area)*
Identifies a buffer containing the order expression to be used for this operation.

For details on how to use an order expression with the SPECIFY VIEW command, see Developing CICSPlex SM applications.

Note: You cannot specify the EYU_CICSNAME or EYU_CICSREL attributes in an order expression.

LENGTH*(data-value)*
A fullword value that specifies the length of the FIELDS buffer.

Note: The buffer length you specify should not include any data other than an order expression.

OBJECT*(data-value)*
Identifies the resource table for which a view is being created. This value must be the 1- to 8-character name of a valid CICS Resource table. For a list of the CICSPlex SM resource tables by type, see Developing CICSPlex SM applications.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

VIEW*(data-value)*
Defines a 1- to 8-character name for the view being built.

Conditions

The following is a list of the RESPONSE values that can be returned by the SPECIFY VIEW command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

DUPE
A duplicate condition occurred for the following reason:

VIEW
The specified view already exists and cannot be built.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected in either the command string or the FIELDS buffer. The parameter that is invalid is returned as the reason value:

- ATTRIBUTE
- FIELDS
- LENGTH
- OBJECT

- THREAD
- VIEW.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

NOTFOUND
A not found condition occurred for the following reason:

ATTRIBUTE
An attribute specified in the FIELDS buffer was not found for the specified resource table.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for one of the following reasons:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if the resource table is missing required attributes, contains one or more conflicting attributes, or does not exist. Use the FEEDBACK command to retrieve additional data about this error.

INVALIDVER
The specified version of the resource table is not supported by CICSPlex SM.

VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

TERMINATE

Terminate all API processing on all active threads.

Description

This command terminates all API processing on all active threads created by the CICS or MVS task that issues the command.

- Issuing TERMINATE is equivalent to issuing the DISCONNECT command for each active thread individually.
- Any resources that are associated with the thread are released, including result sets, filters, views, diagnostic data, and outstanding asynchronous requests.

Related commands

CONNECT, DISCONNECT

Options

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

Conditions

The following is a list of the RESPONSE values that can be returned by the TERMINATE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

FAILED
The command failed for one of the following reasons:

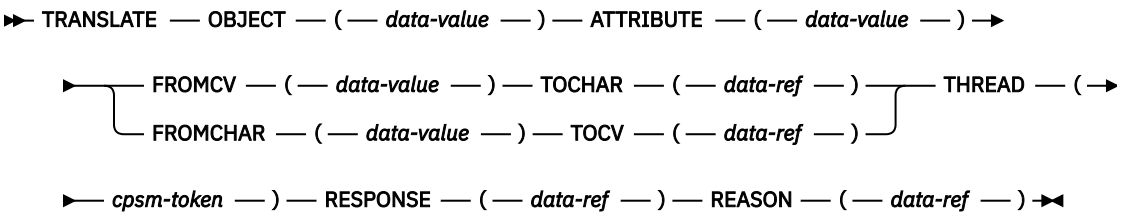
ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

TRANSLATE

Translate resource table attribute values.

Command



Description

This command translates resource table attribute values that are maintained in an encoded form (such as EYUDA and CVDA values) between their internal coded format and an external display format.

- If your program is written in REXX, you can use the TPARSE command to access a resource table record and translate its attribute values. However, if you use the ASIS option with TPARSE, attribute values are not translated into their external format; in that case, you would need to use TRANSLATE after using TPARSE to receive the formatted display values. For a description of the TPARSE command, see Chapter 2, “REXX functions and commands,” on page 11.
- In a CICS environment, the DFHVALUE function can return incompatible CVDA values for the following resource table attributes:

Resource table	Attribute value
CONNECT	RECOVSTATUS(NRS)
IPCONN	RECOVSTATUS(NRS)
LIBRARY	CRITSTATUS(CRITICAL)
LOCTRAN	RESSEC(RESSECEXT)
PROGDEF	API(CICSAPI)
PROGRAM	APIST(CICSAPI)
PROGRAM	LPASTAT(NOTSVA)
PROGRAM	LPASTAT(SVA)

Because these CVDA values conflict with values used in CICS, CICSPlex SM must modify them to retain their uniqueness. CICSPlex SM adds either 8000 or 9000 to the value returned by DFHVALUE for each of these CICS CVDA attributes.

If you want to translate any of these attributes, you must add either 8000 or 9000 to the value you received from DFHVALUE before presenting the attribute to CICSPlex SM.

Options

ATTRIBUTE*(data-value)*
Identifies the resource table attribute that is to be translated. This value must be the 1- to 12-character name of a valid attribute for the resource table.

FROMCHAR*(data-value)*
Specifies the 1- to 12-character value for the specified attribute.

FROMCV*(data-value)*
Specifies the 4-byte internal coded value for the specified attribute.

OBJECT*(data-value)*
Identifies the resource table to which the attribute being translated belongs. This value must be the 8-character name of a valid resource table.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

TOCHAR*(data-ref)*
Names a variable to receive the result of translating an internal coded value to the 1- to 12-character value for the specified attribute.

TOCV*(data-ref)*
Names a variable to receive the result of translating a character value to the 4-byte internal coded value for the specified attribute.

Conditions

The following is a list of the RESPONSE values that can be returned by the TRANSLATE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- ATTRIBUTE
- FROMCHAR
- FROMCV
- OBJECT
- THREAD
- TOCHAR
- TOCV.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for the following reason:

INVALIDVER
The specified version of the resource table is not supported by CICSPlex SM.

VERSIONINVL
A version conflict occurred for one of the following reasons:

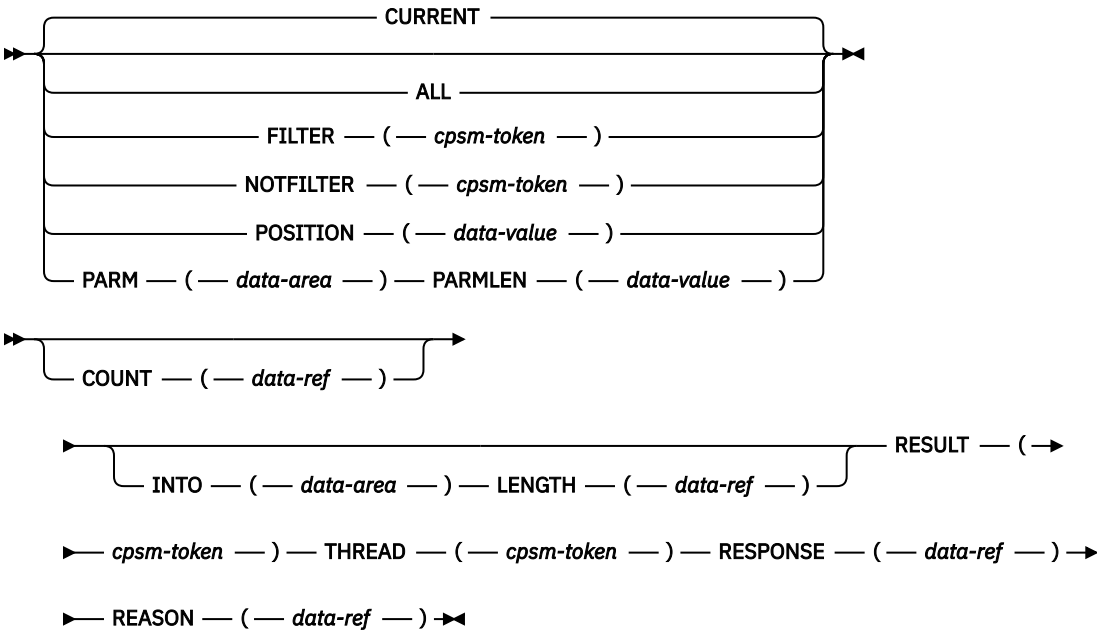
NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

UNMARK

Remove the marks placed on resource table records.

records_group



Description

This command removes the marks placed on resource table records by a previous MARK command. The UNMARK command always begins processing with the last record that was fetched, rather than the next one in the result set.

Related commands

EXPAND, LOCATE, MARK

Options

ALL
Removes the marks from all resource table records in the result set.

COUNT*(data-ref)*
Names a variable to receive the number of resource table records that could not be unmarked.

CURRENT
Removes the mark from only the current resource table record.

FILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The FILTER option removes the marks from only those resource table records that meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

INTO*(data-area)*
Identifies a buffer to receive a list of resource table records that could not be unmarked.

This buffer must be long enough to hold the maximum number of record numbers that could result from your UNMARK request (in the event that none of them can be unmarked). Record numbers are listed individually (not by range) in the INTO buffer and are separated by commas.

Note: If you receive a RESPONSE value of WARNING AREATOOSMALL (because the buffer was not long enough), the data returned in this buffer represents a partial list of the records that could not be unmarked.

LENGTH*(data-ref)*
A fullword value that specifies the length of the INTO buffer.

The value that CICSPlex SM returns in this field depends on the RESPONSE value for the UNMARK command:

OK
The actual length of the data returned in the INTO buffer.

WARNING AREATOOSMALL
The buffer length that would be required to hold a complete list of records that could not be unmarked.

NOTFILTER*(cpsm-token)*
Identifies a filter to be used for this operation. The NOTFILTER option removes the marks from only those resource table records that do not meet the specified filter criteria.

The *cpsm-token* value that identifies a filter is returned by the SPECIFY FILTER command.

PARM*(data-area)*
Identifies a buffer containing the parameter expression that lists the resource table records to be unmarked.

The parameter expression for the UNMARK command is a character string of record numbers. For example:

```
PARM('1,3,6:9,24.')
```

To specify individual records, separate the record numbers with a comma. To specify a range of records, separate the low and high record numbers with a colon. The whole parameter expression must end with a period.

For details on how to use a parameter expression with the UNMARK command, see Developing CICSPlex SM applications.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

POSITION*(data-value)*
Removes the mark from the nth resource table record in the result set.

This value must be a number that identifies the record's relative position in the result set. The first record in a result set is identified by the number 1.

For example, to unmark the fifth resource table record in a result set, you would specify `POSITION(5)`.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set can be one produced by any of these commands:

- COPY
- GET
- GETDEF
- GROUP
- PERFORM OBJECT.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the UNMARK command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

NODATA
No records were found that matched the specified search criteria.

WARNING
The command completed processing with a warning, for one of the following reasons:

AREATOOSMALL
You specified the INTO and LENGTH options, but the buffer was not long enough to hold the string of records that could not be unmarked.

DATAERROR
One or more of the records specified in the PARM buffer could not be found to be unmarked. If you specified the COUNT option, the number of records that could not be unmarked is returned. If you specified the INTO and LENGTH options, a list of the records is returned in the buffer.

BUSY
A busy condition occurred for the following reason:

RESULT
The result set specified on the RESULT option is being processed by another command.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

SOLRESOURCE
A required resource that is locally owned (that is, owned by the address space where the processing thread is running) is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- COUNT

- FILTER
- INTO
- LENGTH
- NOTFILTER
- PARM
- PARMLEN
- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

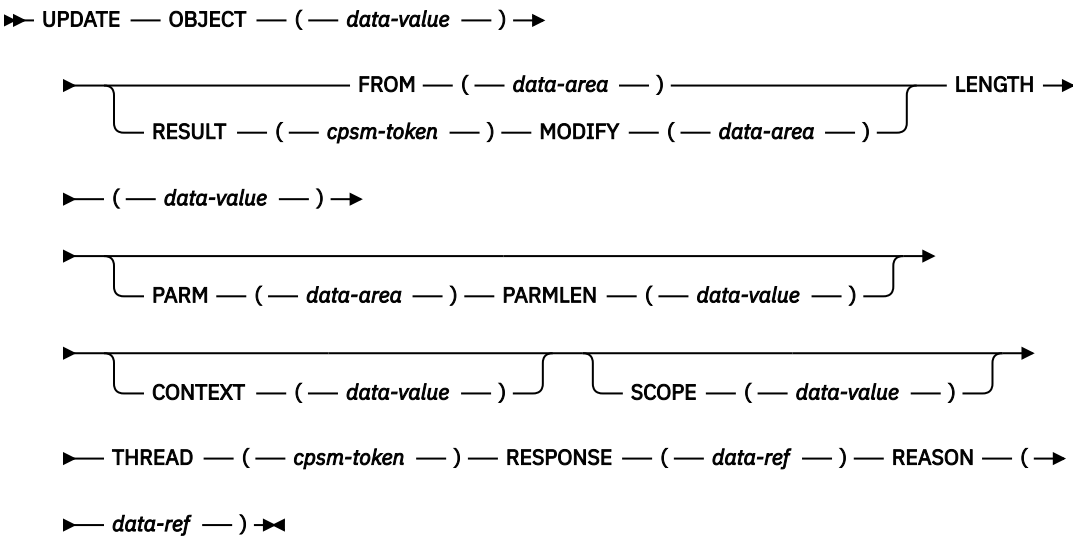
VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

UPDATE

Update an existing CICSPlex SM or CICS definition.



Description

This command updates an existing CICSPlex SM or CICS definition according to the attribute values you specify.

- The updated definition replaces the existing definition in the CICSPlex SM data repository.
- For definitions that have a CICSplex as their context (such as workload management or real-time analysis definitions), the definition is also updated in the data repositories of all CMASs involved in managing the CICSplex.

Related commands

CREATE, REMOVE

Options

CONTEXT*(data-value)*
Identifies the context for this command. The context must be the 1- to 8-character name of a CMAS or CICSplex.

If you do not specify the CONTEXT option, the default context for the thread is assumed.

FROM*(data-area)*
Identifies a buffer containing a resource table record that represents the definition to be updated.

The record must include all of the attributes for the resource table specified on the OBJECT option. For optional attributes that you do not want to specify, set the field to null (that is, zero) values .

LENGTH*(data-value)*
A fullword value that specifies the length of the FROM or MODIFY buffer.

Note: The buffer length you specify should not include any data other than a resource table record or modification expression.

MODIFY*(data-area)*
Identifies a buffer containing the modification expression to be used in modifying CICS Definition resource table records.

For details on how to form a modification expression, see Developing CICSPlex SM applications.

OBJECT*(data-value)*
Identifies the resource table that represents the definition being updated. This value must be the 8-character name of a valid CICSPlex SM Definition or CICS Definition resource table. For a list of the CICSPlex SM resource tables by type, see Developing CICSPlex SM applications.

PARM*(data-area)*
Identifies a buffer containing a parameter expression to be used in updating the definition.

For details on how to use a parameter expression with the UPDATE command, see Developing CICSPlex SM applications. For a description of the parameters that are valid for a given resource table, see the CICSPlex SM resource tables in Reference.

PARMLEN*(data-value)*
A fullword value that specifies the length of the PARM buffer.

REASON*(data-ref)*
Names a variable to receive the fullword reason value returned by this command.

RESPONSE*(data-ref)*
Names a variable to receive the fullword response value returned by this command.

RESULT*(cpsm-token)*
Identifies the API result set to be processed by this operation. The result set must contain CICS Definition resource table records. The records are updated according to the modification expression you supply in the MODIFY buffer.

The result set can be one produced by any of these commands:

- COPY
- GET

- GROUP
- PERFORM OBJECT.

SCOPE*(data-value)*
Identifies the scope for this command.

If the current context (as set by this command or a previous CONNECT or QUALIFY command) is a CICSplex and the OBJECT option identifies a CICS definitional resource and the PARM option includes the CSD parameter, a valid scope can be specified.

The scope can be a CICS system within the CICSplex. If the current context is a CMAS or the OBJECT option identifies any other type of resource table, or the CSD parameter is not specified on a CICS definitional resource, this option has no meaning and is ignored.

If SCOPE applies to the command and you do not specify the SCOPE option, the default scope for the thread is assumed. If the current context is a CICSplex and no default scope has been set on a CONNECT or QUALIFY command, you receive an INVALIDPARM response for the SCOPE option.

THREAD*(cpsm-token)*
Identifies the API thread to be used for this operation. The *cpsm-token* value that identifies a thread is returned by the CONNECT command.

Conditions

The following is a list of the RESPONSE values that can be returned by the UPDATE command. The description of each RESPONSE includes a list of associated REASON values, if appropriate.

OK
The command completed processing successfully.

ENVIRONERROR
An environment error occurred for one of the following reasons:

NOSERVICE
The application stub program could not load the API service module.

NOSTORAGE
The application stub program could not obtain the necessary storage in the address space where the processing thread is running.

NOTPROCESSED
For CSD requests only, one of the MASs to which the request was directed could not process the request.

REQTIMEOUT
One of the CMASs or MASs to which the request was directed did not respond.

SOCRESOURCE
A required resource that is owned by the CMAS is not available.

FAILED
The command failed for one of the following reasons:

ABENDED
Command processing abended.

EXCEPTION
Command processing encountered an exceptional condition.

INVALIDATA
An invalid data error occurred for one of the following reasons:

MODIFY
An attribute value listed in the MODIFY buffer is not valid for the specified attribute.

NOTSUPPORTED
An attribute listed in the MODIFY buffer is not modifiable.

INVALIDCMD
The command is invalid for the following reason:

LENGTH
The total length of all the options on the command exceeds the maximum limit.

INVALIDPARM
An invalid parameter was detected. The parameter that is invalid is returned as the reason value:

- CONTEXT
- FROM
- LENGTH
- MODIFY
- OBJECT
- PARM
- PARMLEN
- RESULT
- THREAD.

Check the command description for valid parameter syntax.

NOTAVAILABLE
A not available condition occurred for one of the following reasons:

APITASK
The API control subtask is not active.

CMAS
A CMAS to which the request was directed is not available.

CPSMAPI
The CMAS to which the processing thread is connected is not available for API processing.

CSDAPI
Support for the CICSPlex SM API to access the CICS CSD is not available.

MAINTPOINT
The maintenance point for the current context is not available.

NOTPERMIT
A not permitted condition occurred for the following reason:

USRID
The user ID associated with the processing thread does not have the required security authorization.

SERVERGONE
The CMAS to which the processing thread was connected is no longer active.

TABLEERROR
A resource table record is invalid for one of the following reasons:

DATAERROR
The value associated with one or more resource table attributes is invalid. This error can occur if:

- The resource table is missing required attributes, contains one or more conflicting attributes, or does not exist.
- A CICS resource definition contains attributes that would cause the EXEC CICS CREATE command to issue warnings.

Use the FEEDBACK command to retrieve additional data about this error.

INVALIDATTR
One of the resource table attributes is invalid.

INVALIDVER
The specified version of the resource table is not supported by CICSPlex SM.

VERSIONINVL
A version conflict occurred for one of the following reasons:

NOTSUPPORTED
The version of the application stub program used for this command is not supported.

NOTVSNCONN
The version of the application stub program used for this command is not the same as the version used with the CONNECT command.

Chapter 4. CICSPlex SM API command argument list

The CICSPlex SM API command arguments are listed.

For each EXEC CPSM command, the CICS translator inserts a call to the CICSPlex SM EXEC interface stub program. See Link editing your program.

Parameters of the EXEC CPSM command are passed with the standard call parameter list; the list of EXEC CPSM command arguments, ARG0 - ARG*n*:

Argument 0
ARG0 contains command-specific data. The first two bytes of ARG0 contain the function code of the CPSM command. For a list of function code values, see Chapter 5, “CICSPlex SM API command function code values,” on page 125.

Argument 1
ARG1 contains the THREAD value. For the TERMINATE command, ARG1 contains zeros.

Argument 2
ARG2 contains the OBJECT value, when applicable. For a list of commands that can contain a valid OBJECT value, see Chapter 5, “CICSPlex SM API command function code values,” on page 125.

Argument 3
ARG3 contains command-specific data.

Argument 4
ARG4 contains command-specific data.

Argument 5
ARG5 returns the response code. See Chapter 6, “RESPONSE and REASON values,” on page 129.

Argument 6
ARG6 returns the reason code. See Chapter 6, “RESPONSE and REASON values,” on page 129.

Argument 7 and higher
ARG7 and higher contain command-specific data.

Chapter 5. CICSPlex SM API command function code values

The CICSPlex SM API commands and their numeric function code values are listed.

Table 1 on page 125 is ordered by command name, and Table 2 on page 126 is ordered by function code value.

For information about the arguments, see Chapter 4, "CICSPlex SM API command argument list," on page 123.

Table 1. Function code values ordered by command name

CICSPlex SM command	Number of arguments	Function Code	OBJECT value in ARG2
ADDRESS	7	F016	No
CANCEL	10	F002	No
CONNECT	13	F003	No
COPY	11	F004	No
CREATE	10	F005	Yes
DELETE	7	F006	No
DISCARD	7	F007	No
DISCONNECT	7	F008	No
EXPAND	16	F021	No
FEEDBACK	10	F020	No
FETCH	10	F009	No
GET	7	F00A	Yes
GETDEF	7	F017	Yes
GROUP	12	F01E	Yes
LISTEN	14	F018	No
LOCATE	7	F00B	No
MARK	7	F00C	No
ORDER	10	F00D	No
PERFORM OBJECT	10	F00F	No
PERFORM SET	10	F00E	Yes
QUALIFY	7	F010	No
QUERY	7	F011	No
RECEIVE	7	F012	Yes
REFRESH	7	F019	No
REMOVE	10	F013	Yes
SET	10	F01A	No

Table 1. Function code values ordered by command name (continued)			
CICSPlex SM command	**Number of arguments**	**Function Code**	**OBJECT value in ARG2**
SPECIFY FILTER	11	F014	Yes
SPECIFY VIEW	11	F01B	Yes
TERMINATE	7	F01C	No
TRANSLATE	13	F01D	Yes
UNMARK	7	F015	No
UPDATE	10	F01F	Yes

Table 2. Function code values ordered by function code			
Function Code	**Number of arguments**	**CICSPlex SM command**	**OBJECT value in ARG2**
F002	10	CANCEL	No
F003	13	CONNECT	No
F004	11	COPY	No
F005	10	CREATE	Yes
F006	7	DELETE	No
F007	7	DISCARD	No
F008	7	DISCONNECT	No
F009	10	FETCH	No
F00A	7	GET	Yes
F00B	7	LOCATE	No
F00C	7	MARK	No
F00D	10	ORDER	No
F00E	10	PERFORM SET	Yes
F00F	10	PERFORM OBJECT	No
F010	7	QUALIFY	No
F011	7	QUERY	No
F012	7	RECEIVE	Yes
F013	10	REMOVE	Yes
F014	11	SPECIFY FILTER	Yes
F015	7	UNMARK	No
F016	7	ADDRESS	No
F017	7	GETDEF	Yes
F018	14	LISTEN	No
F019	7	REFRESH	No
F01A	10	SET	No

Table 2. Function code values ordered by function code (continued)			
Function Code	**Number of arguments**	**CICSPlex SM command**	**OBJECT value in ARG2**
F01B	11	SPECIFY VIEW	Yes
F01C	7	TERMINATE	No
F01D	13	TRANSLATE	Yes
F01E	12	GROUP	Yes
F01F	10	UPDATE	Yes
F020	10	FEEDBACK	No
F021	16	EXPAND	No

Chapter 6. RESPONSE and REASON values

This section provides a summary of the RESPONSE and REASON values returned by each API command.

For descriptions of these values, refer to the description of the command that returns them. For a list of RESPONSE and REASON character values and their numeric equivalents, see Chapter 7, “EYUDA values,” on page 137. For a discussion of the RESPONSE and REASON options, see Developing CICSPlex SM applications.

COMMAND	RESPONSE	REASONS
ADDRESS		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	ECB, SENTINEL, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
CANCEL		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	NOTIFICATION, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
CONNECT		
	OK	
	ENVIRONERROR	APITASKERR, INVALIDTCB, NOSERVICE, NOSTORAGE, SOCRESOURCE, SOERESOURCE, SOLRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	CONTEXT, SCOPE, SIGNONPARM, USRID, VERSION
	NOTAVAILABLE	APITASK, CPSMAPI, CPSMSERVER, CPSMSYSTEM, CPSMVERSION
	NOTPERMIT	USRID
	VERSIONINVL	NOTSUPPORTED
COPY		
	OK	
	NODATA	
	BUSY	FROM, TO
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INCOMPATIBLE	INVALIDOBJ
	INVALIDPARM	FILTER, FROM, NOTFILTER, THREAD, TO
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	TABLEERROR	INVALIDVER

COMMAND	RESPONSE	REASONS
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
CREATE		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDCMD	LENGTH
	INVALIDPARM	CONTEXT, FROM, LENGTH, OBJECT, PARM, PARMLEN, THREAD
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, CSDAPI, MAINTPOINT
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR, INVALIDATTR, INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
DELETE		
	OK	
	NODATA	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE, SOLRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	FILTER, NOTFILTER, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
DISCARD		
	OK	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INUSE	FILTER, VIEW
	INVALIDPARM	FILTER, RESULT, THREAD, VIEW
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
DISCONNECT		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
FEEDBACK		
	OK	
	NODATA	
	WARNING	AREATOOSMALL

COMMAND	RESPONSE	REASONS
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	COUNT, INTO, LENGTH, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
FETCH		
	OK	
	NODATA	BACKWARD, FORWARD
	WARNING	AREATOOSMALL
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE, SOLRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	COUNT, FILTER, INTO, LENGTH, NOTFILTER, POSITION, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
GET		
	OK	
	SCHEDULED	
	NODATA	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDATA	CRITERIA
	INVALIDCMD	FILTER,LENGTH
	INVALIDPARM	CONTEXT, CRITERIA, FILTER, LENGTH, OBJECT, PARM, PARMLEN, RESULT, SCOPE, THREAD, TOKEN
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, CSDAPI, MAINTPOINT, SCOPE, WORKLOAD
	NOTFOUND	ATTRIBUTE
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR, INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
GETDEF		
	OK	
	NODATA	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INCOMPATIBLE	INVALIDOBJ
	INVALIDPARM	ATTRIBUTE, OBJECT, RESOURCE, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI

COMMAND	RESPONSE	REASONS
	SERVERGONE	
	TABLEERROR	DATAERROR, INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
GROUP		
	OK	
	NODATA	
	BUSY	FROM, TO
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDCMD	LENGTH
	INVALIDPARM	BY, FILTER, FROM, LENGTH, NOTFILTER, SUMOPT, THREAD, TO
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	TABLEERROR	INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
LISTEN		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INCOMPATIBLE	INVALIDEVT
	INVALIDPARM	CONTEXT, EVENT, FILTER, NOTFILTER, NOTIFICATION, THREAD, TOKEN
	NOTAVAILABLE	APITASK, CPSMAPI, PLEXMGR
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
LOCATE		
	OK	
	NODATA	BACKWARD, FORWARD
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE, SOLRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	BACKWARD, FILTER, FORWARD, NOTFILTER, POSITION, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
MARK		
	OK	
	NODATA	
	WARNING	AREATOOSMALL, DATAERROR
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE, SOLRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	COUNT, FILTER, INTO, LENGTH, NOTFILTER, PARM, PARMLEN, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	

COMMAND	RESPONSE	REASONS
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
ORDER		
	OK	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDCMD	LENGTH
	INVALIDPARM	BY, LENGTH, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
PERFORM OBJECT		
	OK	
	SCHEDULED	
	NODATA	BACKWARD, FORWARD
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDATA	PARM, CRITERIA
	INVALIDCMD	FILTER,LENGTH
	INVALIDPARM	ACTION, CONTEXT, CRITERIA, FILTER, LENGTH, OBJECT, PARM, PARMLEN, RESULT, SCOPE, THREAD, TOKEN
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, CSDAPI, MAINTPOINT, PLEXMGR, SCOPE, WORKLOAD
	NOTFOUND	ACTION, ATTRIBUTE
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
	WARNING	RESULT, ACTION
PERFORM SET		
	OK	
	SCHEDULED	
	NODATA	BACKWARD, FORWARD
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDATA	PARM, CRITERIA
	INVALIDCMD	LENGTH
	INVALIDPARM	ACTION, FILTER, NOTFILTER, PARM, PARMLEN, POSITION, RESULT, THREAD, TOKEN
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, MAINTPOINT, PLEXMGR, SCOPE, WORKLOAD
	NOTFOUND	ACTION, ATTRIBUTE
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR

COMMAND	RESPONSE	REASONS
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
QUALIFY		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	CONTEXT, SCOPE, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI, PLEXMGR
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
QUERY		
	OK	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	CONTEXT, DATALENGTH, OBJECT, RESULT, THREAD, TYPE
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
RECEIVE		
	OK	
	NODATA	
	WARNING	AREATOOSMALL
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	DELAY, INTO, LENGTH, OBJECT, THREAD, TOKEN
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
REFRESH		
	OK	
	SCHEDULED	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	COUNT, FILTER, NOTFILTER, RESULT, THREAD, TOKEN
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, MAINTPOINT, SCOPE
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
REMOVE		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION

COMMAND	RESPONSE	REASONS
	INVALIDCMD	LENGTH
	INVALIDPARM	CONTEXT, FROM, LENGTH, OBJECT, PARM, PARMLEN, THREAD
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, CSDAPI, MAINTPOINT
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR, INVALIDATTR, INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
SET		
	OK	
	SCHEDULED	
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDATA	MODIFY, NOTSUPPORTED
	INVALIDCMD	LENGTH
	INVALIDPARM	ATTRIBUTE, COUNT, FILTER, LENGTH, MODIFY, NOTFILTER, POSITION, RESULT, THREAD, TOKEN
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, MAINTPOINT, SCOPE
	NOTFOUND	ACTION, ATTRIBUTE
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR, INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
SPECIFY FILTER		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDCMD	FILTER,LENGTH
	INVALIDPARM	CRITERIA, FILTER, LENGTH, OBJECT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	NOTFOUND	ATTRIBUTE
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
SPECIFY VIEW		
	OK	
	DUPE	VIEW
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDATA	CRITERIA
	INVALIDCMD	LENGTH
	INVALIDPARM	ATTRIBUTE, FIELDS, LENGTH, OBJECT, THREAD, VIEW
	NOTAVAILABLE	APITASK, CPSMAPI
	NOTFOUND	ATTRIBUTE
	SERVERGONE	
	TABLEERROR	DATAERROR, INVALIDVER

COMMAND	RESPONSE	REASONS
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
TERMINATE		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE
	FAILED	ABENDED, EXCEPTION
TRANSLATE		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDPARM	ATTRIBUTE, FROMCHAR, FROMCV, OBJECT, THREAD, TOCHAR, TOCV
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	TABLEERROR	INVALIDVER, INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
UNMARK		
	OK	
	NODATA	
	WARNING	AREATOOSMALL, DATAERROR
	BUSY	RESULT
	ENVIRONERROR	NOSERVICE, NOSTORAGE, SOCRESOURCE, SOLRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDCMD	LENGTH
	INVALIDPARM	COUNT, FILTER, INTO, LENGTH, NOTFILTER, PARM, PARMLEN, RESULT, THREAD
	NOTAVAILABLE	APITASK, CPSMAPI
	SERVERGONE	
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN
UPDATE		
	OK	
	ENVIRONERROR	NOSERVICE, NOSTORAGE, NOTPROCESSED, REQTIMEOUT, SOCRESOURCE
	FAILED	ABENDED, EXCEPTION
	INVALIDCMD	LENGTH
	INVALIDPARM	CONTEXT, FROM, LENGTH, MODIFY, OBJECT, PARM, PARMLEN, RESULT, THREAD
	NOTAVAILABLE	APITASK, CMAS, CPSMAPI, CSDAPI, MAINTPOINT
	NOTPERMIT	USRID
	SERVERGONE	
	TABLEERROR	DATAERROR, INVALIDATTR, INVALIDVER
	VERSIONINVL	NOTSUPPORTED, NOTVSNCONN

Chapter 7. EYUDA values

This section lists the CICSPlex SM API EYUDA values and their numeric equivalents.

There are three types of EYUDAs:

General
Values that CICSPlex SM uses to describe or define a resource. These EYUDAs have numeric values in the range of 1– 776. See “EYUDA general values in numerical order” on page 137.

RESPONSE
Values returned by the RESPONSE option of an API command. These EYUDAs have numeric values in the range of 1024 – 1042. See “EYUDA RESPONSE values in numerical order” on page 167.

REASON
Values returned by the REASON option of an API command. These EYUDAs have numeric values in the range of 1280 – 1378. See “EYUDA REASON values in numerical order” on page 169.

Note: The EYUDA values and their numeric equivalents listed for SUMMUNLIKE are only returned for an EYUDA attribute in an API GROUP command result set, or a WUI summary tabular view. In previous releases this was set to N_a.

Note: The EYUDA values and their numeric equivalents listed for HOTPOOL are valid only in CICS Transaction Server 2.3.

EYUDA general values in numerical order

This section lists the general EYUDAs in numerical order and shows the character value for each.

Value	EYUDA
0	N_A
0	NA
1	YES
2	NO
3	ON
4	OFF
5	VALID
6	INVALID
7	TRUE
8	FALSE
9	VLS
10	LS
11	LW
12	NM
13	HW
14	HS
15	VHS
16	EQ

Value	EYUDA
17	NE
18	GT
19	LT
20	LOW
21	HIGH
22	NORMAL
23	IMMEDIATE
24	TAKEOVER
25	SHUT
26	NOSHUT
27	GLOBAL
28	SYSTEM
30	SUSPEND
33	VALUE
34	THRESHOLD
35	SAM
36	APM
37	MRM
38	TRANID
39	TERMID
40	SIGNID
41	RACFGID
42	USERID
43	NULL
44	CHARSTR
45	ACTIVE
46	INACTIVE
47	WAITING
48	QUIESCING
49	POOL
50	LTRAN
51	RTRAN
53	GOAL
54	QUEUE
55	LUNAME

Value	EYUDA
58	DELIMIT
59	PCONV
60	LOGON
61	SIGNON
63	PERMANENT
64	MCICS
65	MGLBL
66	MDBX
67	MCONN
68	MFILE
69	MJRNL
70	MPROG
71	MTERM
72	MTDQS
73	MTRAN
74	MAPPL
75	ABOVE
76	BELOW
77	NOCOPY
78	DSA
79	CDSA
80	UDSA
81	LPA
82	EDSA
83	ECDSA
84	EUDSA
85	ERDSA
86	ELPA
87	CICS
88	USER
89	READONLY
90	LU61
91	LU62
92	INDIRECT
93	MRO

Value	EYUDA
94	NOTAPPLIC
95	LFILE
96	RFILE
97	CTABL
98	UTABL
99	INSTALLED
100	PENDING
101	INHERIT
102	EXPLICIT
103	CICSSYS
104	SYSGROUP
105	KEEP
106	NAME
107	FORCE
108	NONE
109	UNASSIGNED
110	DROP
111	LOCAL
112	REMOTE
113	DEFAULT
114	REMOVE
115	DORMANT
116	START
117	END
118	ADJACENT
119	LOSTCON
120	CREATING
121	REMOVING
122	QUIESCED
123	LINKACTIVE
124	LINKDOWN
125	ESSS
126	CONACT
127	RESET
128	SYSDUMP

Value	EYUDA
129	TRANDUMP
130	MAXTASK
131	STALLED
132	SOSUDSA
133	SOSCDSA
134	SOSEUDSA
135	SOSECDSA
136	SOSERDSA
137	SOSSDSA
138	SOSESDSA
139	SOSRDSA
140	QUIESCE
141	PRIMARY
142	SECONDARY
143	DUPLICATE
144	FROZEN
145	ALL
146	ANY
147	SUM
148	MIN
149	MAX
150	AVG
151	CNT
152	LE
153	GE
154	SDSA
155	ESDSA
156	RDSA
157	SOSMVS
158	SOSBELOW
159	SOSABOVE
161	EXECUTE
162	CHECK
163	LOSTCMAS
164	LOSTMAS

Value	EYUDA
165	AASTERISK
166	BLANK
167	INDEX
168	DATA
169	BOTH
170	NETBIOS
171	TCPIP
172	AFTER
173	ALLREQS
174	ASA
177	ASIS
178	ASSEMBLER
179	BACKOUTONLY
180	BEFORE
181	BLUE
182	C
183	CLEARCONV
184	CLOSE
185	COBOL
186	COLD
187	CYCLIC
188	DEFERRED
189	DIP
190	DISK
191	EODS
192	EXTA
193	FILE
194	FIRSTREF
195	GREEN
196	IDENTIFY
197	IGNORE
198	INITIAL
199	INOUT
200	INPUT
201	LEAVE

Value	EYUDA
202	LE370
203	LINEAR
204	LINK
205	LMS
206	LOGICAL
207	LOGOFF
208	LRU
209	MACHINE
210	MESSAGE
211	MIXIDPE
212	MOD
213	MODIFYREQS
214	MSRE
215	NEUTRAL
216	NEW
217	NOFORCE
218	NONVTAM
219	OLD
220	ONLY
221	OPEN
222	OPID
223	OUTPUT
224	PERSISTENT
225	PHYSICAL
226	PINK
227	PLI
228	PRINTER
229	RED
230	RECOVERY
231	REJECT
232	RELEASESESS
233	REREAD
234	RPG
235	SCS
236	SECURITY

Value	EYUDA
237	SHR
238	SKIP
239	SPECIFIC
240	STARTIO
241	STARTUP
242	STRFIELD
243	SYSDEFAULT
244	TAPE
245	TERMINAL
246	TRANSACTION
247	TRANSIENT
248	TURQUOISE
249	U
250	UNCONDREL
251	UPDATEONLY
252	VB
253	VERIFY
254	VTAM
255	YELLOW
256	3270
257	AUTO
258	DYNAM
259	EXTRA
260	INTRA
261	IND
262	STAT
263	RELATED
264	TARGET
265	NEVER
266	ALWAYS
267	COLDONLY
268	WARMONLY
269	PROMPT
270	CONTINUE
271	TERMINATE

Value	EYUDA
272	SHUTDOWN
273	RTADEF
274	STATDEF
275	CONNDEF
276	FILEDEF
277	JRNLDEF
278	JRNMDEF
279	LSRDEF
280	MAPDEF
281	PARTDEF
282	PRTNDEF
283	PROFDEF
284	PROGDEF
285	SESSDEF
286	TDQDEF
287	TERMDEF
288	TRANDEF
289	TRNCLDEF
290	TSQDEF
291	TYPTMDEF
292	MPSYNCCR
293	ASSOCIATIONS
294	MEMBERS
295	DB2CDEF
296	DB2EDEF
297	DB2TDEF
298	FSEGDEF
299	TSMDEF
300	ENQMDEF
301	TCPDEF
302	DOCDEF
303	FULL
304	RELEASE
305	PA1
306	PA2

Value	EYUDA
307	PA3
308	PF1
309	PF2
310	PF3
311	PF4
312	PF5
313	PF6
314	PF7
315	PF8
316	PF9
317	PF10
318	PF11
319	PF12
320	PF13
321	PF14
322	PF15
323	PF16
324	PF17
325	PF18
326	PF19
327	PF20
328	PF21
329	PF22
330	PF23
331	PF24
332	STANDARD
333	APPC
334	BATCHDI
335	BCHLU
336	CONSOLE
337	CONTLU
338	INTLU
339	LUTYPE2
340	LUTYPE3
341	LUTYPE4

Value	EYUDA
342	L3277
343	L3284
344	L3286
345	PIPELINE
346	SCSPRINT
347	TLX
348	TWX
349	USERPROG
350	3270P
351	3275
352	3277
353	3277CM
354	3284
355	3284CM
356	3286
357	3286CM
358	3600
359	3614
360	3650
361	3653
362	3767
363	3767C
364	3767I
365	3770
366	3770B
367	3770C
368	3770I
369	3790
370	BLINK
371	REVERSE
372	UNDERLINE
373	INSTALL
374	RESDEF
375	RASINDSC
376	RESTYPE

Value	EYUDA
377	SCOPETYP
378	REBUILD
379	RECONNECT
380	CONNECTING
381	TWAIT
382	NOTWAIT
383	DISCONNING
384	CFTBL
385	CF
386	QUASIRENT
387	THREADSAFE
388	PROCDEF
389	BAPPL
390	ACTIVITY
391	PROCESS
393	PMWINDOW
394	FULLSCREEN
395	COM1
396	COM2
397	COM3
398	COM4
399	COM5
400	COM6
401	COM7
402	COM8
403	CLIENTAUTH
404	RQMDEF
405	3270TERM
406	3270PRNT
407	3270DBTM
408	3270DBPR
409	3151TERM
410	SEQTERM
411	EOF
412	EOT

Value	EYUDA
413	FEPOODEF
414	FETRGDEF
415	FENODDEF
416	FEPRODEF
417	DEBUG
418	LEVSE
419	KEY
420	RBA
423	EJCODEF
424	EJDJDEF
425	BASIC
426	CERTIFICATE
427	AUTOREGISTER
428	AUTOMATIC
429	CLIENTCERT
430	LINK3270
431	FACILITY
432	NOTOPEN
433	UNKNOWN
434	REENTPROT
435	NOREENTPROT
436	ASSERTED
437	MIRROR
438	DPL
439	ONCRPC
440	WEB
441	BRIDGE
442	CICSBTS
443	TDQUEUE
444	TERMSTART
445	XMRUN
446	SOCKET
447	RRS
448	IIRQRECVR
449	RZSTTRPT

Value	EYUDA
450	IIOP
451	ECI
452	HOTPOOL
453	JVM
454	HTASK
455	IOERROR
456	INVALIDFILE
457	SUSPENDED
458	SUSPENDING
459	HISTORY
460	REALTIME
461	VELOCITY
462	DISCRETIONRY
463	CONNECTED
464	NOTCONNECTED
465	RESUMING
466	URIMPDEF
467	PIPEDEF
468	WEBSVDEF
469	STARTED
470	STOPPED
471	XPLINK
472	SSL
473	CICSAPI
474	OPENAPI
475	EXCI
476	RECREATED
477	IPCONDEF
478	SAME
479	SUPPORTED
480	SOSGCDSA
481	JCL
482	REGION
483	IEFUSI
484	SMF

Value	EYUDA
485	ABOVEBAR
486	LIBDEF
487	ENABLED
488	DISABLED
489	SNA
490	IPIC
491	PARTIAL
492	ATOMDEF
493	BUNDDEF
494	MQCONDEF
495	JVMSVDEF
496	THREADED
497	CSDLIST
498	CSDGROUP
499	CSDINLST
500	CSDINGRP
501	LOCKED
502	UOW
503	SIT
504	EYU
505	TABLE
506	SYSIN
507	WUI
508	REQUIRED
509	COMBINED
510	LNQUEUE
511	LNGOAL
512	GROUPRESYNC
513	INSTALLING
514	DISCARDING
566	SUMMUNLIKE
567	ATTLSAWARE
568	MQMON
640	MODBNOTACTIVE
641	PRIVATEMETHOD

Value	EYUDA
642	METHODLOCKED
643	ENVIRONTIMEOUT
644	ENVIRONCANCEL
645	STARTFAILED
646	STACKTOOLARGE
647	OVERFLOWOSSBFAIL
648	BADMALTRANSFER
736	TRANSMITTIMEOUT
737	EXECUTETIMEOUT
738	TRANSMITFAILED
739	TARGETUNAVAIL
740	SECVIOLATION
741	MALRETURNFAILED
742	MAXHISTORYRECS
743	REQNOTPROCED
752	NOTAUTHORIZED
753	SECINTERFACEFAIL
754	TRUSTED
755	ETDSA
756	EMPTY
757	INCOMPLETE
758	INVALIDSCOPE
759	DISABLING
760	ENABLING
761	SOMEDISABLED
762	BYTE
763	KILOBYTE
764	MEGABYTE
765	GIGABYTE
766	THOUSAND
767	SECOND
768	MILLISECOND
769	MICROSECOND
770	ABEND
771	PLATFORM

Value	EYUDA
772	APPLICATION
773	APPLMAJVER
774	APPLMINVER
775	APPLMICVER
776	OPERATION
777	IMPORTONLY
778	AVAILABLE
779	UNAVAILABLE
780	SOMEAVAIL
781	520
782	MQMONDEF
783	TASK

EYUDA general values in alphabetical order

This section lists the general EYUDAs in alphabetical order by their character values.

EYUDA	Value
AASTERISK	165
ABEND	770
ABOVE	75
ABOVEBAR	485
ACTIVE	45
ACTIVITY	390
ADJACENT	118
AFTER	172
ALL	145
ALLREQS	173
ALWAYS	266
ANY	146
APM	36
APPC	333
APPLICATION	772
APPLMAJVER	773
APPLMICVER	775
APPLMINVER	774
ASA	174

EYUDA	Value
ASIS	177
ASSEMBLER	178
ASSERTED	436
ASSOCIATIONS	293
ATOMDEF	492
AUTO	257
AUTOMATIC	428
AUTOREGISTER	427
AVAILABLE	778
AVG	150
BACKOUTONLY	179
BAPPL	389
BASIC	425
BATCHDI	334
BCHLU	335
BEFORE	180
BELOW	76
BLANK	166
BLINK	370
BLUE	181
BOTH	169
BRIDGE	441
BUNDDEF	493
BYTE	762
C	182
CDSA	79
CERTIFICATE	426
CF	385
CFTBL	384
CHARSTR	44
CHECK	162
CICS	87
CICSAPI	473
CICSBTS	442
CICSSYS	103

EYUDA	Value
CLEARCONV	183
CLIENTAUTH	403
CLIENTCERT	429
CLOSE	184
CNT	151
COBOL	185
COLD	186
COLDONLY	267
COM1	395
COM2	396
COM3	397
COM4	398
COM5	399
COM6	400
COM7	401
COM8	402
CONACT	126
CONNDEF	275
CONNECTED	463
CONNECTING	380
CONSOLE	336
CONTINUE	270
CONTLU	337
CREATING	120
CSDLIST	497
CSDGROUP	498
CSDINLST	500
CTABL	97
CYCLIC	187
DATA	168
DB2CDEF	295
DB2EDEF	296
DB2TDEF	297
DEBUG	417
DEFAULT	113

EYUDA	Value
DEFERRED	188
DELIMIT	58
DIP	189
DISABLED	488
DISCONNING	383
DISCRETIONARY	462
DISK	190
DOCDEF	302
DORMANT	115
DPL	438
DROP	110
DSA	78
DUPLICATE	143
DYNAM	258
ECDSA	83
ECI	451
EDSA	82
EJCODEF	423
EJDJDEF	424
ELPA	86
ENABLED	487
END	117
ENQMDEF	300
EODS	191
EOF	411
EOT	412
EQ	16
ERDSA	85
ESDSA	155
ESSS	125
EUDSA	84
EXECUTE	161
EXPLICIT	102
EXTA	192
EXTRA	259

EYUDA	Value
FACILITY	431
FALSE	8
FENODDEF	415
FEPOODEF	413
FEPRODEF	416
FETRGDEF	414
FILE	193
FILEDEF	276
FIRSTREF	194
FORCE	107
FROZEN	144
FSEGDEF	298
FULL	303
FULLSCREEN	394
GE	153
GIGABYTE	765
GLOBAL	27
GOAL	53
GREEN	195
GT	18
HIGH	21
HISTORY	459
HOTPOOL	452
HS	14
HTASK	454
HW	13
IDENTIFY	196
IEFUSI	483
IGNORE	197
IIOP	450
IIRQRECVR	448
IMMEDIATE	23
INACTIVE	46
IND	261
INDEX	167

EYUDA	Value
INDIRECT	92
INHERIT	101
INITIAL	198
INOUT	199
INPUT	200
INSTALL	373
INSTALLED	99
INTLU	338
INTRA	260
INVALID	6
INVALIDFILE	456
IOERROR	455
IPCONDEF	475
JCL	481
JRNLDEF	277
JRNMDEF	278
JVM	453
JVMVDEF	495
KEEP	105
KEY	419
KILOBYTE	763
LE	152
LEAVE	201
LEVSE	418
LE370	202
LFILE	95
LIBDEF	486
LINEAR	203
LINK	204
LINKACTIVE	123
LINKDOWN	124
LINK3270	430
LMS	205
LNGOAL	511
LNQUEUE	510

EYUDA	Value
LOCAL	111
LOCKED	501
LOGICAL	206
LOGOFF	207
LOGON	60
LOSTCMAS	163
LOSTCON	119
LOSTMAS	164
LOW	20
LPA	81
LRU	208
LS	10
LSRDEF	279
LT	19
LTRAN	50
LUNAME	55
LUTYPE2	339
LUTYPE3	340
LUTYPE4	341
LU61	90
LU62	91
LW	11
L3277	342
L3284	343
L3286	344
MACHINE	209
MAPDEF	280
MAPPL	74
MAX	149
MAXTASK	130
MCICS	64
MCONN	67
MDBX	66
MEGABYTE	764
MEMBERS	294

EYUDA	Value
MESSAGE	210
MFILE	68
MGLBL	65
MICROSECOND	769
MILLISECOND	768
MIN	148
MIRROR	437
MIXIDPE	211
MJRNL	69
MOD	212
MODIFYREQS	213
MPROG	70
MPSYNCCR	292
MQCONDEF	494
MRM	37
MRO	93
MSRE	214
MTDQS	72
MTERM	71
MTRAN	73
N_A	0
NA	0
NAME	106
NE	17
NETBIOS	170
NEUTRAL	215
NEVER	265
NEW	216
NM	12
NO	2
NOCOPY	77
NOFORCE	217
NONE	108
NONVTAM	218
NOREENTPORT	435

EYUDA	Value
NORMAL	22
NOSHUT	26
NOTAPPLIC	94
NOTCONNECTED	464
NOTOPEN	432
NOTWAIT	382
NULL	43
OFF	4
OLD	219
ON	3
ONCRPC	439
ONLY	220
OPEN	221
OPENAPI	474
OPERATION	776
OPID	222
OUTPUT	223
PARTDEF	281
PA1	305
PA2	306
PA3	307
PCONV	59
PENDING	100
PERMANENT	63
PERSISTENT	224
PF1	308
PF10	317
PF11	318
PF12	319
PF13	320
PF14	321
PF15	322
PF16	323
PF17	324
PF18	325

EYUDA	Value
PF19	326
PF2	309
PF20	327
PF21	328
PF22	329
PF23	330
PF24	331
PF3	310
PF4	311
PF5	213
PF6	313
PF7	314
PF8	315
PF9	316
PHYSICAL	225
PINK	226
PIPEDEF	467
PIPELINE	435
PLATFORM	771
PLI	227
PMWINDOW	393
POOL	49
PRIMARY	141
PRINTER	228
PROCDEF	388
PROCESS	391
PROFDEF	283
PROGDEF	284
PROMPT	269
PRTNDEF	282
QUASIRENT	386
QUEUE	54
QUIESCE	140
QUIESCED	122
QUIESCING	48

EYUDA	Value
RACFGID	41
RASINDSC	375
RBA	420
RDSA	156
READONLY	89
REALTIME	460
REBUILD	378
RECONNECT	379
RECOVERY	230
RED	229
REENTPROT	434
REGION	482
REJECT	231
RELATED	263
RELEASE	304
RELEASESESS	232
REMOTE	112
REMOVE	114
REMOVING	121
REREAD	233
RESDEF	374
RESET	127
RESTYPE	376
RESUMING	465
REVERSE	371
RFILE	96
RPG	234
RQMDEF	404
RRS	447
RTADEF	273
RTRAN	51
RZSTTRPT	449
SAM	35
SCOPETYP	337
SCS	235

EYUDA	Value
SCSPRINT	346
SDSA	154
SECOND	767
SECONDARY	142
SECURITY	236
SEQTERM	410
SESSDEF	285
SHR	237
SHUT	25
SHUTDOWN	272
SIGNID	40
SIGNON	61
SKIP	238
SMF	484
SOCKET	446
SOMEAVAIL	780
SOSABOVE	159
SOSBELOW	158
SOSCDSA	133
SOSECDSA	135
SOSERDSA	136
SOSESDSA	138
SOSEUDSA	134
SOSGCDSA	480
SOSMVS	157
SOSRDSA	139
SOSSDSA	137
SOSUDSA	132
SPECIFIC	239
SSL	472
STALLED	131
STANDARD	332
START	116
STARTED	332
STARTIO	240

EYUDA	Value
STARTUP	241
STAT	262
STATDEF	274
STOPPED	470
STRFIELD	242
SUM	147
SUMMUNLIKE	566
SUSPEND	30
SUSPENDED	457
SUSPENDING	458
SYSDEFAULT	243
SYSDUMP	128
SYSGROUP	104
SYSTEM	28
TAKEOVER	24
TAPE	244
TARGET	264
TCPDEF	301
TCPIP	171
TDQDEF	286
TDQUEUE	443
TERMDEF	287
TERMID	39
TERMINAL	245
TERMINATE	271
TERMSTART	444
THOUSAND	766
THREADED	496
THREADSAFE	387
THRESHOLD	34
TLX	347
TRANDEF	288
TRANDUMP	129
TRANID	38
TRANSACTION	246

EYUDA	Value
TRANSIENT	247
TRNCLDEF	289
TRUE	7
TSMDEF	299
TSQDEF	290
TURQUOISE	248
TWAIT	381
TWX	348
TYPTMDEF	291
U	249
UDSA	80
UNASSIGNED	109
UNAVAILABLE	779
UNCONDREL	250
UNDERLINE	372
UNKNOWN	433
UOW	502
UPDATEONLY	251
URIMPDEF	466
USER	88
USERID	42
USERPROG	349
UTABL	98
VALID	5
VALUE	33
VB	252
VELOCITY	461
VERIFY	352
VHS	15
VLS	9
VTAM	254
WAITING	47
WARMONLY	268
WEB	440
WEBSVDEF	468

EYUDA	Value
XMRUN	445
XPLINK	471
YELLOW	255
YES	1
3151TERM	409
3270	256
3270DBPR	408
3270DBTM	407
3270P	350
3270PRNT	406
3270TERM	405
3275	351
3277	352
3277CM	353
3284	354
3284CM	355
3286	356
3286CM	357
3600	358
3614	359
3650	360
3653	361
3767	362
3767C	363
3767I	364
3770	365
3770B	366
3770C	367
3770I	368
3790	369

EYUDA RESPONSE values in numerical order

This section lists the RESPONSE EYUDAs in numerical order.

Value	EYUDA
1024	OK

Value	EYUDA
1025	SCHEDULED
1026	NOTFOUND
1027	NODATA
1028	INVALIDPARM
1029	FAILED
1030	ENVIRONERROR
1031	NOTPERMIT
1032	BUSY
1033	SERVERGONE
1034	NOTAVAILABLE
1035	VERSIONINVL
1036	INVALIDCMD
1037	WARNING
1038	TABLEERROR
1039	INCOMPATIBLE
1040	INUSE
1041	INVALIDATA
1042	DUPE

EYUDA RESPONSE values in alphabetical order

This section lists the RESPONSE EYUDAs in alphabetical order.

Value	EYUDA
BUSY	1032
DUPE	1042
ENVIRONERROR	1030
FAILED	1029
INCOMPATIBLE	1039
INUSE	1040
INVALIDATA	1041
INVALIDCMD	1036
INVALIDPARM	1028
NODATA	1027
NOTAVAILABLE	1034
NOTFOUND	1026
NOTPERMIT	1031

Value	EYUDA
OK	1024
SCHEDULED	1025
SERVERGONE	1033
TABLEERROR	1038
VERSIONINVL	1035
WARNING	1037

EYUDA REASON values in numerical order

This section lists the REASON EYUDAs in numerical order.

Value	EYUDA
1280	THREAD
1281	OBJECT
1282	CONTEXT
1283	RESULT
1284	COUNT
1285	LENGTH
1286	FILTER
1287	NOTFILTER
1288	FORWARD
1289	BACKWARD
1290	POSITION
1291	DELAY
1292	NOTIFICATION
1293	SIGNONPARM
1294	SCOPE
1295	RESOURCE
1296	FROM
1297	TO
1298	INTO
1299	CRITERIA
1300	BY
1301	ACTION
1302	ECB
1303	SENTINEL
1304	FEEDBACK

Value	EYUDA
1305	EVENT
1306	TOKEN
1307	MODIFY
1308	VIEW
1309	FIELDS
1310	ATTRIBUTE
1311	FROMCV
1312	TOCHAR
1313	FROMCHAR
1314	TOCV
1315	PARM
1316	PARMLEN
1317	SUMOPT
1318	TYPE
1319	DATALENGTH
1320	SOLRESOURCE
1321	SOCRESOURCE
1322	SOERESOURCE
1323	MAINTPOINT
1324	SYSNOTACT
1325	SYSLVLBAD
1326	SYSNOTLCL
1327	CICSRELBAD
1328	ARMNOTREG
1329	ARMNOTACT
1330	ARMPOLCHK
1331	ABENDED
1332	CPSMSYSTEM
1333	CPSMVERSION
1334	CPSMAPI
1335	NOTSUPPORTED
1336	NOTVSNCONN
1337	INVALIDATTR
1338	APITASKERR
1339	CPSMSERVER

Value	EYUDA
1340	APITASK
1341	PLEXMGR
1342	REQTIMEOUT
1344	AREATOOSMALL
1345	USRID
1348	VERSION
1352	FILTERMATCH
1353	INVALIDOBJ
1354	INVALIDVER
1355	TASKDATAKEY
1356	INVALIDVERB
1357	NOSTORAGE
1358	NOSERVICE
1359	EXCEPTION
1360	INVALIDEVT
1361	DATAERROR
1362	CMAS
1363	FIRST
1364	NEXT
1365	EXPIRED
1366	WORKLOAD
1367	ACTIONPARM
1368	CICSNAME
1369	MAXRECORDS
1370	QUERY
1371	EXPAND
1372	XINTO
1373	XLENGTH
1374	APICMD
1375	CSDAPI
1376	TEST
1377	QUERYERROR
1378	NOTPROCESSED
1379	INVALIDTCB

EYUDA REASON values in alphabetical order

This section lists the REASON EYUDAs in alphabetical order.

Value	EYUDA
ABENDED	1331
ACTION	1301
ACTIONPARM	1367
APITASK	1340
APITASKERR	1338
AREATOOSMALL	1344
ARMNOTACT	1329
ARMNOTREG	1328
ARMPOLCHK	1330
ATTRIBUTE	1310
BACKWARD	1289
BY	1300
CICSNAME	1368
CICSRELBAD	1327
CMAS	1362
CONTEXT	1282
COUNT	1284
CPSMAPI	1334
CPSMSERVER	1339
CPSMSYSTEM	1332
CPSMVERSION	1333
CRITERIA	1299
CSDAPI	1375
DATAERROR	1361
DATALENGTH	1319
DELAY	1291
ECB	1302
EVENT	1305
EXCEPTION	1359
EXPIRED	1365
FEEDBACK	1394
FIELDS	1309
FILTER	1286

Value	EYUDA
FITLERMATCH	1352
FIRST	1363
FORWARD	1288
FROM	1296
FROMCHAR	1313
FROMCV	1311
INTO	1298
INVALIDATTR	1337
INVALIDEVT	1360
INVALIDOBJ	1353
INVALIDTCB	1379
INVALIDVER	1354
INVALIDVERB	1356
LENGTH	1285
MAINTPOINT	1323
MAXRECORDS	1369
MODIFY	1307
NEXT	1364
NOSERVICE	1358
NOSTORAGE	1357
NOTFILTER	1287
NOTIFICATION	1292
NOTPROCESSED	1378
NOTSUPPORTED	1335
NOTVSNCONN	1336
OBJECT	1281
PARM	1315
PARMLEN	1316
PLEXMGR	1341
POSITION	1290
QUERY	1370
REQTIMEOUT	1342
RESOURCE	1295
RESULT	1283
SCOPE	1294

Value	EYUDA
SENTINEL	1303
SIGNONPARM	1293
SOCRESOURCE	1321
SOERESOURCE	1322
SOLRESOURCE	1320
SUMOPT	1317
SYSLVLBAD	1325
SYSNOTACT	1324
SYSNOTLCL	1326
TASKDATAKEY	1355
THREAD	1280
TO	1297
TOCHAR	1312
TOCV	1314
TOKEN	1306
TYPE	1318
USRID	1345
VERSION	1348
VIEW	1308
WORKLOAD	1366
XINTO	1372
XLENGTH	1373

Notices

This information was developed for products and services offered in the United States of America. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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 rights 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 grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

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

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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 jurisdictions 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who want 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 Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119 Armonk,
NY 10504-1785
United States of America

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

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.

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 actual people or business enterprises 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Programming interface information

IBM CICS supplies some documentation that can be considered to be Programming Interfaces, and some documentation that cannot be considered to be a Programming Interface.

Programming Interfaces that allow the customer to write programs to obtain the services of CICS Transaction Server for z/OS, Version 5 Release 5 (CICS TS 5.5) are included in the following sections of the online product documentation:

- Developing applications
- Developing system programs
- CICS security
- Developing for external interfaces
- Reference: application development
- Reference: system programming
- Reference: connectivity

Information that is NOT intended to be used as a Programming Interface of CICS TS 5.5, but that might be misconstrued as Programming Interfaces, is included in the following sections of the online product documentation:

- Troubleshooting and support
- Reference: diagnostics

If you access the CICS documentation in manuals in PDF format, Programming Interfaces that allow the customer to write programs to obtain the services of CICS TS 5.5 are included in the following manuals:

- Application Programming Guide and Application Programming Reference
- Business Transaction Services

- Customization Guide
- C++ OO Class Libraries
- Debugging Tools Interfaces Reference
- Distributed Transaction Programming Guide
- External Interfaces Guide
- Front End Programming Interface Guide
- IMS Database Control Guide
- Installation Guide
- Security Guide
- CICS Transactions
- CICSPlex System Manager (CICSPlex SM) Managing Workloads
- CICSPlex SM Managing Resource Usage
- CICSPlex SM Application Programming Guide and Application Programming Reference
- Java™ Applications in CICS

If you access the CICS documentation in manuals in PDF format, information that is NOT intended to be used as a Programming Interface of CICS TS 5.5, but that might be misconstrued as Programming Interfaces, is included in the following manuals:

- Data Areas
- Diagnosis Reference
- Problem Determination Guide
- CICSPlex SM Problem Determination Guide

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Apache, Apache Axis2, Apache Maven, Apache Ivy, the Apache Software Foundation (ASF) logo, and the ASF feather logo are trademarks of Apache Software Foundation.

Gradle and the Gradlephant logo are registered trademark of Gradle, Inc. and its subsidiaries in the United States and/or other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Red Hat®, and Hibernate® are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

Spring Boot is a trademark of Pivotal Software, Inc. in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Zowe™, the Zowe logo and the Open Mainframe Project™ are trademarks of The Linux Foundation.

The Stack Exchange name and logos are trademarks of Stack Exchange Inc.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability
These terms and conditions are in addition to any terms of use for the IBM website.

Personal use
You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use
You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights
Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM online privacy statement

IBM Software products, including software as a service solutions, (*Software Offerings*) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information (PII) is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect PII. If this Software Offering uses cookies to collect PII, specific information about this offering's use of cookies is set forth below:

For the CICSPlex SM Web User Interface (main interface):
Depending upon the configurations deployed, this Software Offering may use session and persistent cookies that collect each user's user name and other PII for purposes of session management, authentication, enhanced user usability, or other usage tracking or functional purposes. These cookies cannot be disabled.

For the CICSPlex SM Web User Interface (data interface):
Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's user name and other PII for purposes of session management, authentication, or other usage tracking or functional purposes. These cookies cannot be disabled.

For the CICSPlex SM Web User Interface ("hello world" page):
Depending upon the configurations deployed, this Software Offering may use session cookies that do not collect PII. These cookies cannot be disabled.

For CICS Explorer®:

Depending upon the configurations deployed, this Software Offering may use session and persistent preferences that collect each user's user name and password, for purposes of session management, authentication, and single sign-on configuration. These preferences cannot be disabled, although storing a user's password on disk in encrypted form can only be enabled by the user's explicit action to check a check box during sign-on.

If the configurations deployed for this Software Offering provide you, as customer, the ability to collect PII from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM Privacy Policy and IBM Online Privacy Statement, the section entitled *Cookies, Web Beacons and Other Technologies* and the IBM Software Products and Software-as-a-Service Privacy Statement.

Index

O

P

Q

R

S

T

U