

# Elastic Transaction Platform

## Programmer's Guide

REVISION: FEBRUARY 2015

The contents of this manual may be revised without prior notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the expressed written permission of Heirloom Computing Inc.

Heirloom Computing has made every effort to ensure that this manual is correct and accurate, but reserves the right to make changes without notice at its sole discretion at any time.

# Preface

---

This Programmer's Guide provides guidelines for the usage of Elastic Transaction Platform (ETP).

Elastic Transaction Platform is a robust enterprise sub-system permitting CICS based COBOL applications to migrate and execute within the Java Enterprise Edition (JEE) platform.

Elastic Transaction Platform integrates business applications into a cohesive, flexible, enterprise solution that allows existing business processes to use and exploit Java Enterprise based technologies. Elastic Transaction Platform reduces time, complexity, and risk involved in building and redeploying CICS COBOL business applications. The Elastic Transaction Platform preserves existing business assets, reduces the total cost of ownership and increases flexibility as offered through Enterprise Java Bean (EJB) transactions.

Elastic Transaction Platform provides traditional business sub-system services in the Java Enterprise Application Server environment. These sub-system services will function in conjunction with the popular application servers available on a wide variety of operating platforms. The application infrastructure permits enterprises to integrate business applications, and use an integrated environment built using industry standards.

# Trademarks

---

- IBM is a registered trademark of International Business Machines Inc.
- Oracle and Java are registered trademarks of Oracle and/or its affiliates.
- UNIX is a registered trademark licensed exclusively to X/Open Company Limited.
- Linux is a registered trademark of Linus Torvalds.
- Windows is a registered trademark of Microsoft Corporation.
- Eclipse is a trademark of the Eclipse Foundation Inc.
- Other names may be trademarks of their respective owners.

# Contents

---

<b>ELASTIC TRANSACTION PLATFORM USAGE.....</b>	<b>5</b>
JOURNALS.....	5
JOURNAL PROTOCOLS .....	6
TEMPLATES.....	9
QUEUES .....	10
QUEUE PROTOCOLS.....	10
SQL .....	12
FILE DEFINITIONS .....	13
<b>BMS FIELD VALIDATION EXTENSIONS.....</b>	<b>16</b>
FMASK .....	16
NMASK .....	18
EVENT.....	19
HTMLINPUT.....	20
<b>ETP COMMAND SYNTAX .....</b>	<b>21</b>
ABEND .....	22
ADDRESS.....	23
ADDRESS SET .....	25
ASKTIME .....	25
ASSIGN .....	26
BIF DEEDIT .....	36
CANCEL.....	38
CHANGE PASSWORD .....	38
CHANGE TASK.....	39
DEFINE COUNTER .....	40
DELAY .....	42
DELETE.....	44
DELETE COUNTER .....	45
DELETEQ TD.....	47
DELETEQ TS .....	47
DOCUMENT CREATE .....	48
DEQ .....	50
DOCUMENT INSERT .....	51
DOCUMENT RETRIEVE .....	54
DOCUMENT SET.....	55
DUMP TRANSACTION .....	56
ENDBR .....	58
ENQ .....	59
ENTER TRACENUM .....	61
EXTRACT CERTIFICATE .....	62
EXTRACT TCPIP .....	64
EXTRACT WEB.....	66
FORMATTIME .....	66
FREEMAIN .....	69
GET COUNTER.....	69
GETMAIN.....	72
JOURNAL .....	73

HANDLE ABEND .....	73
HANDLE AID .....	74
HANDLE CONDITION .....	75
IGNORE CONDITION .....	76
LINK .....	76
LOAD .....	78
MONITOR .....	80
POP HANDLE .....	80
POST .....	81
PURGE MESSAGE .....	83
PUSH HANDLE .....	83
QUERY COUNTER .....	84
QUERY SECURITY .....	85
READ .....	87
READNEXT .....	90
READPREV .....	92
READQ TD .....	95
READQ TS .....	96
RECEIVE MAP .....	98
RECEIVE PARTN .....	99
RELEASE .....	100
RESETBR .....	100
RETRIEVE .....	102
RETURN .....	104
REWIND COUNTER .....	105
REWRITE .....	106
ROUTE .....	108
SEND CONTROL .....	110
SEND MAP .....	111
SEND PAGE .....	113
SEND PARTNSET .....	114
SEND TEXT .....	114
SIGNOFF .....	116
SIGNON .....	116
SPOOLCLOSE .....	118
SPOOLOPEN INPUT .....	119
SPOOLOPEN OUTPUT .....	120
SPOOLREAD .....	122
SPOOLWRITE .....	123
START .....	124
SUSPEND .....	128
START ATTACH .....	128
START BREXIT .....	129
STARTBR .....	130
SUSPEND .....	132
SYNCPOINT .....	133
SYNCPOINT ROLLBACK .....	133
TRACE .....	134
UNLOCK .....	135
UPDATE COUNTER .....	136
UNLOCK .....	138
VERIFY PASSWORD .....	139
WAIT EXTERNAL .....	140

WAIT EVENT .....	141
WAIT JOURNALNAME.....	142
WAIT JOURNALNUM .....	142
WAITCICS.....	143
WAIT JOURNALNAME.....	144
WAIT JOURNALNUM .....	145
WEB ENDBROWSE COOKIE .....	146
WEB ENDBROWSE FORMFIELD .....	146
WEB ENDBROWSE HTTPHEADER.....	147
WEB EXTRACT .....	147
WEB READ COOKIE .....	151
WEB READ FORMFIELD.....	152
WEB READ HTTPHEADER .....	154
WEB READNEXT COOKIE .....	155
WEB READNEXT FORMFIELD .....	156
WEB READNEXT HTTPHEADER.....	157
WEB RECEIVE.....	158
WEB RETRIEVE.....	159
WEB SEND .....	160
WEB STARTBROWSE COOKIE.....	162
WEB STARTBROWSE FORMFIELD .....	162
WEB STARTBROWSE HTTPHEADER.....	163
WEB WRITE COOKIE.....	163
WEB WRITE HTTPHEADER .....	165
WRITE .....	166
WRITE JOURNALNAME .....	168
WRITE JOURNALNUM.....	169
WRITE JOURNALNAME .....	171
WRITE OPERATOR .....	172
WRITEQ TD .....	174
WRITEQ TS.....	176
XCTL.....	177
<b>APPENDIX DFHVALUE(NAME) .....</b>	<b>179</b>
DFHVALUE CONSTANTS .....	179
DFHVALUE(NAME) DEVICE COMMANDS .....	195
<b>APPENDIX DFHRESP .....</b>	<b>197</b>
<b>APPENDIX EJB SETUP FILES.....</b>	<b>200</b>
<b>APPENDIX UNSUPPORTED COMMANDS.....</b>	<b>201</b>
DISTRIBUTED PROCESSING SERVICE: .....	201
INTERVAL CONTROL:.....	201
BATCH DATA INTERCHANGE:.....	201
TCP/IP:.....	201
TRACE CONTROL: .....	201
TASK MANAGEMENT SERVICE: .....	2

# Elastic Transaction Platform Usage

---

When compiling for the Elastic Transaction Platform (ETP) with Elastic COBOL, the compiler option `'-out:transaction'` must be included. Otherwise, EXEC TRANSACTION or EXEC CICS statements will be not be recognized.

The following resources must be configured so that ETP can be used:

- Journals
- Templates
- Queues
- SQL
- File Definitions

## Journals

---

The journal protocol mechanism is used by several services, including Journal, Operator, and Trace.

A journal protocol provides a structured mechanism for writing to a sequential data store. It may optionally provide asynchronous writes and a reply mechanism. Note that not all journal protocols are available from all environments. For instance, while a Dialog journal may be appropriate for an operator message in an application, writing to the Dialog journal using Journal commands in an EJB would at best, pause services, and at worst, not function at all.

The protocol's class implementation is defined using the setup value `protocol.journal.name=classname`. The default values for the protocols are available in the table below. A protocol that does not support reply will produce condition EXPIRED:100, while a protocol that does not support wait will ignore it.

Protocol	Supports		Default Value
	Reply	Wait	
protocol.journal.file	N	Y	com.heirloomcomputing.etp.transaction.journal.BinaryFile
protocol.journal.textfile	N	Y	com.heirloomcomputing.etp.transaction.journal.TextFile
protocol.journal.weblogic	N	N	com.heirloomcomputing.etp.transaction.journal.WebLogic
protocol.journal.log4j	N	N	com.heirloomcomputing.etp.transaction.journal.Log4J
protocol.journal.sysout	N	Y	com.heirloomcomputing.etp.transaction.journal.Sysout

protocol.journal.syserr	N	Y	com.heirloomcomputing.etp.transaction.journal.Syserr
protocol.journal.dialog	Y	N	com.heirloomcomputing.etp.transaction.journal.Dialog
protocol.journal.queue	N	Y	com.heirloomcomputing.etp.transaction.journal.Queue
protocol.journal.binary	N	Y	com.heirloomcomputing.etp.transaction.journal.BinaryFile
protocol.journal.text	N	Y	com.heirloomcomputing.etp.transaction.journal.TextFile
protocol.journal.bea	N	N	com.heirloomcomputing.etp.transaction.journal.WebLogic
protocol.journal.apache	N	N	com.heirloomcomputing.etp.transaction.journal.Log4J
protocol.journal.stdout	N	Y	com.heirloomcomputing.etp.transaction.journal.Sysout
protocol.journal.stderr	N	Y	com.heirloomcomputing.etp.transaction.journal.Syserr

## Journal Protocols

---

All Journal Protocols are in lower-case in setup. Each is followed by a colon (:), and a referent. The referent is further information specific to the protocol.

In general, the Journal service uses the setup entry 'journal.journalname'. The Operator service uses the setup entry 'operator.#' where # is the routing number (2 by default). The Trace service uses the setup entry 'trace'.

### Example:

```
journal.myjournal=sysout:
operator.2=myjournal
trace=myjournal
```

Protocols intended for logging destinations will interpret certain JTYPEID codes specially. The following JTYPEID codes are interpreted special wherever applicable (case insensitive):

JTYPEID	Interpretation
*D	Debug
*I	Information
*W	Warning
*E	Error
*F	Fatal Error
*T	Trace
!C	Operator Critical Level
!I	Operator Immediate Level
!E	Operator Eventual Level
!R	Operator Reply Required

## file:

The 'binary:' protocol is also available.

The file: protocol appends its entries to a binary file. The referent is the filename.

The format of the binary entry is:

Field Name	Size
Length of entry	4 bytes
User(0) / System(1)	4 bytes
Task Number	4 bytes
Transaction ID	4 bytes
Terminal ID	4 bytes
Timestamp (milliseconds)	8 bytes
JTYPEID	2 bytes
Length of user prefix	4 bytes
Length of user data	4 bytes
User Prefix	Bytes given by length of user prefix
User Data	Bytes given by length of user data

Example setup (sends output for journal 'mybin' to the file 'testmyfile.bin'):

```
journal.mybin=file:testmyfile.bin
```

## textfile:

The 'text:' protocol.

The textfile: protocol appends its entries to a textfile.

The format of the text entry is two lines:

```
[JTYPEID] user-prefix: date-time (EIBTASKN taskn) (EIBTRMID term-id)  
(EIBTRNSID trans-id):
```

```
user-data
```

Example setup (sends output for journal 'mytxt' to the file 'testmyfile.txt'):

```
journal.mytxt=file:testmyfile.txt
```

## weblogic:

The 'bea:' protocol.

This protocol is available only when operating under Oracle WebLogic.

The weblogic: protocol sends its entries to the WebLogic logging mechanism. In the WebLogic console, click the domain name in the navigator, then 'View domain log' on the right for a brief view of the logged messages. The full message file is available as well.

The format is determined by the current format of WebLogic output.

Example setup (sends output for journal 'myweb' to WebLogic catalog 'testmyfile.weblogic'):

```
journal.myweb=weblogic:testmyfile.weblogic
```

### **log4j:**

The 'apache:' protocol.

The Log4J protocol sends its output to an Apache Log4J logger. The referent is the logger name.

Apache Log4J has its own setup mechanisms and is generally not included by default with the application server. Its jar file (such as log4j-1.2.7.jar) must be added to either the application server itself or to the application. As this is a general service, we recommend adding it to the application server itself if possible.

The Log4J package itself will search for the file log4j.properties.

In addition to this default action, the setup property 'log4.properties' may be set to a configuration file that will be read when the logger is opened. Only use this if Log4J will be included with the application rather than the application server.

### **sysout:**

The 'stdout:' protocol.

The sysout protocol sends its output directly to the system output stream. This is often the console for an application server.

Example setup (sends output for journal 'mysysout' to sysout):

```
journal.mysysout=sysout:
```

### **syserr:**

The 'stderr:' protocol.

The syserr protocol sends its output directly to the system error stream. This is often the console for an application server.

Example setup (sends output for journal 'mysyserr' to syserr):

```
journal.mysyserr=syserr:
```

### **dialog:**

Dialog sends its messages to a dialog box. Dialog does support replies, however it is generally not allowed under an application server environment.

Example setup (sends output for 'mydialog' to a dialog box):

```
journal.mydialog=dialog:
```

## queue:

The queue protocol redirects the journal entry as a message through to a queuing protocol. The referent is the symbolic queue name, which in turn must be defined as a queue name in the queuing protocol format.

Example setup (sends output for 'myqueue' to the queue queueimpl):

```
journal.myqueue=queue:queueimpl
```

## Templates

---

Templates are HTML files where all the information is not necessarily present statically. They are stored as resources, and referenced by their name. Templates may be buffered in memory for faster access.

The template must be setup using the syntax `template.name=resource`, where `resource` is the actual resource filename.

Most text within a template is passed directly through to the destination. However, certain commands (related to SSI) are supported.

```
<--#include file="filename_OR_template_name"-->
<--#include virtual="filename_OR_template_name"-->
<--#include template="template_name"-->
<--#include resource="filename"-->
```

The include commands include another template file in the processing on the server. The text of the include command itself is removed from the destination stream, replaced by the contents of the resource.

The include template command looks up another template in the setup by `template_name`, referring to the resource indirectly.

The include resource command looks up the resource directly.

The file or virtual command looks up the resource as both template and directly as a resource, using whichever it finds; this is slightly slower than specifying the form directly.

```
<--#set var="name" value="value"-->
```

A template file has variables, the contents of which may be set and echoed. The `DOCUMENT SET` and `SYMBOLLIST` options also set variables.

The set command sets a variable to a value; it will maintain that value within the template until set again. It cannot, however, override the contents of a variable already set by the `DOCUMENT SET` command.

```
<--#echo var="name"-->
&name;
```

Echo the contents of the variable to the destination. The variable may have been set using `DOCUMENT SET` or `SYMBOLLIST` options or set using the set command.

The & form is more convenient, but be aware that it overlaps with existing entity references of HTML (such as &lt; for a less-than sign). An unrecognized & echo command will be passed through for further interpretation by later stages, such as a web browser, allowing &lt; to work as expected so long as no variable has the same name.

## Queues

---

Queues hold data in sequential order, generally adding items to one end of the queue and retrieving them from the other end.

The Elastic Transaction Platform has a generalized queue mechanism, available for the Transient Data Queue, Temporary Storage Queue and Spool services. A queue is defined using a name and uri, where the uri includes the protocol specifying the implementation of the queue. For example, some queues are available only to the current facility's session and other ones may be available even outside of the Transaction Platform.

Temporary and Transient queues may be written to and read from immediately. Spools must be opened, used, and closed.

The most general form for setup of a Transient or Temporary queue is:

```
queue[.sysid_value].name=uri  
[sysid.name=sysid_value]
```

where the uri is protocol:name, protocol specifying the implementation of the queue, and name being passed to that protocol for further interpretation.

A spool queue for output is defined as:

```
spool[.userid[.class[.node]]]=uri
```

and a spool queue for input as:

```
spool[.userid[.class]]=uri
```

where the queue may be as well specified as desired.

## Queue Protocols

---

### **session:destination**

The session queue protocol has its scope only within the existing session (conversation or pseudo-conversation), tied to a single terminal. It is never shared between simultaneous tasks and it is safe to use exclusively without the use of ENQ and DEQ.

This is the default queue protocol, if only the destination name is specified without a protocol. An instance of a session queue is returned by TQ commands using a queue not listed in the setup table.

### **jms:destination**

The JMS queue uses Java Messaging Service for the queue. It uses any available JMS implementation, such as Oracle MessageQ or MQ-Series / WebSphere MQ. Its scope is defined by the JMS setup for the application server, and it can span tasks, machines, clusters and more. The destination is the JMS destination.

It uses additional table entries, each optional.

`destination.factory=jndi_factory_name`

`destination.user=my_user_name`

`destination.password=my_password`

If no factory is specified, then the default factory is used, if specified:

`jms.default.factory=jndi_factory_name`

### **jmstqt:destination**

The JMS temporary queue with task scope uses Java Messaging Service for the queue. It uses any available JMS implementation, such as Oracle MessageQ or MQ-Series / WebSphere MQ. The queue is automatically eliminated at the end of the task. The destination is the JMS destination.

It uses additional table entries, each optional.

`destination.factory=jndi_factory_name`

`destination.user=my_user_name`

`destination.password=my_password`

If no factory is specified, then the default factory is used, if specified:

`jms.default.factory=jndi_factory_name`

### **jmstqs:destination**

The JMS temporary queue with session scope uses Java Messaging Service for the queue. It uses any available JMS implementation, such as Oracle MessageQ or MQ-Series / WebSphere MQ. The queue must be deleted to eliminate its storage. The destination is the JMS destination.

It uses additional table entries, each optional.

`destination.factory=jndi_factory_name`

`destination.user=my_user_name`

`destination.password=my_password`

If no factory is specified, then the default factory is used, if specified:

`jms.default.factory=jndi_factory_name`

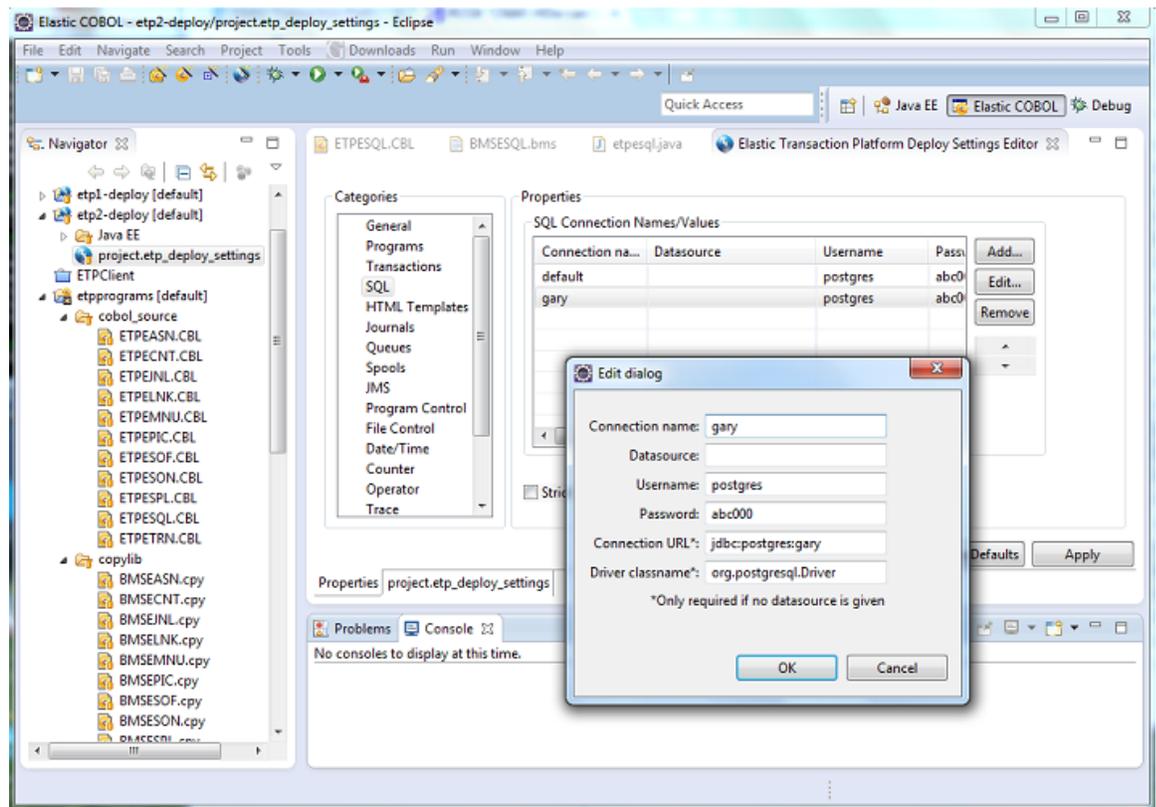
### **null:destination**

The null destination discards its messages.

# SQL

SQL (Structured Query Language) access within the Elastic Transaction Platform must include setup information telling the platform where to find the database connection.

There are two different setup methods, depending on whether or not the SQL database is setup as a datasource in the application server. It is always preferable to setup the datasource if possible. If not supported by the particular application server, specify the SQL connection attributes on the **SQL** tab of the *Elastic Transaction Platform Deploy Settings* within an ETP Deployment Project



Each SQL database connection has its own name. The default name is 'default' and is used if no SET CONNECTION statement is issued. The default connection is automatically established upon first EXEC SQL command by a transaction. You can connect to other databases and even multiple databases and switch the context between them. Use

```
EXEC SQL
    SET CONNECTION { :host-var | conn-name }
END-EXEC
```

Where *host-var* is a PIC X(16) variable defined in the transaction that holds a database connection name or the unquoted *conn-name* that is a database connection name.

The SQL tab of the ETP Deploy Settings file defines the database connection names used by transactions. The editor inserts into the settings property the file following values:

Setup:

```
sql.conn-name.datasource=datasource_name
```

```
sql.conn-name.user=user_name
```

```
sql.conn-name.password=user_password
```

Alternative Setup without datasource:

```
sql.conn-name.url=url_connection_string
```

```
sql.conn-name.driver=class_name
```

```
sql.conn-name.user=user_name
```

```
sql.conn-name.password=user_password
```

The user and/or password may be omitted if supported by the particular database. The *user\_name* is the *user\_name* for connection to the database. The password is only in server setup files and not visible to end clients.

One special SQL connection is of particular note, the 'file' connection. When ETP file I/O is mapped to a relational database (see Elastic COBOL Getting Started Guide) this connection name is used for the file control service commands (READ/READNEXT/READPREV/STARTBR/RESETBR/ENDBR/WRITE/REWRITE).

## File Definitions

---

Elastic Transaction Platform file control is done through SQL. That is, file access traditionally accessed in VSAM datasets is now done through JDBC drivers to SQL database tables.

VSAM data must be converted to SQL using any available VSAM to SQL converter. This may include custom programs written in CICS that read the entire dataset and then write it to SQL, as well as third-party utilities used to migrate data en masse.

The resulting SQL tables must include all the relevant information. FILLER fields need not be maintained if they do not contain data necessary to the containing group.

The mapping from the record structure to the SQL tables may vary significantly; an XML file must be available in the project's resource folder. It must be the same name as the initial VSAM dataset and suffixed with .xml and the setup must include the following entry:

```
file.dataset_name=jdbc:xml_descriptor_file.xml
```

During execution, when the platform accesses the SQL table, it does so using the descriptions given in the XML file. The XML describes how to find

each data element within the record, including its offset, length, and storage format.

Transcribing dataset descriptive information may be tedious and Elastic COBOL also permits the option to place a \$XFD command before the record to be written. The command is `$XFD FILE="filename.xml"`, where the \$ character must be in the indicator column. This command will generate correct information for the COBOL portion of the XML file and a good approximation for the SQL portion. Since the SQL portion can have any column names and storage, the file may need to be hand-edited to match the names used by SQL. After the .xml file is produced, the \$XFD command is no longer needed and may be removed.

The \$XFD command produces the xml descriptive file and places the result in the listing folder. This file must be verified to correct any discrepancies in the actual SQL usage, and the edited file should be moved into the "resources folder". This file is generated to a source folder to prevent accidentally overwriting a hand-edited version of the file.

The root of the XML file is the dataset name. The only entries that matter are the <column> entries. The <group> entries are included for information and for easy usage of group items in the table; the group may be converted to column and its individual columns removed. The column contains the name of the item in the database, the offset, length and type (from *com.heirloomcomputing.ecs.api.Datatype*) in the record, the sqltype for its storage, and the table name in which it is stored. The offset, length and type should not be changed.

The following sqltype values are recognized:

A	ALPHANUMERIC (CHAR)
AV	ALPHANUMERIC VARYING (VARCHAR)
B	BINARY/IMAGE (BINARY)
BV	BINARY/IMAGE VARYING (VARBINARY)
N	NUMERIC INTEGER
NZ	ZONED DECIMAL
N1	SINGLE PRECISION FLOATING POINT
N2	DOUBLE PRECISION FLOATING POINT

The column entry may also include an attribute named `ridfld`, which may be set to true in order to mark it as the primary key. It may also be set to `duplicates` in order to enable duplicate key checks if it is acting as an alternate key. Alternate keys may be set to `noduplicates` in order to suppress duplicate key checks. (The duplicate key check is computationally expensive for databases and should be skipped if the program logic does not require the DUPKEY condition.)

By default, the file control service uses a SQL connection named 'file' to perform its access.

## Example setup:

```
file.alpha=jdbc:alpha.xml
```

## Example data record for reading or writing:

```
*$xfd file="alpha.xml"  
01 my-record.  
  05 surname pic x(20).  
  05 first pic x(20).  
  05 mi pic x.  
  05 ss pic 9(9).  
  05 age pic 9(3).
```

## Example XML dataset definition alpha.xml:

```
<?xml version="1.0" standalone="yes"?>  
  
<dataset name="alpha">  
<!-- This file should be named after your dataset -->  
<group name="my_record" offset="0" length="53" usage="34" sqltype="B" table="alpha">  
<column name="surname" offset="0" length="20" type="32" sqltype="A" table="alpha"/>  
<column name="first" offset="20" length="20" type="32" sqltype="A" table="alpha"  
ridfld="duplicates"/>  
<column name="mi" offset="40" length="1" type="32" sqltype="A" table="alpha"/>  
<column name="ss" offset="41" length="9" type="48" sqltype="NZ" table="alpha"/>  
<column name="age" offset="50" length="3" type="48" sqltype="NZ" table="alpha"/>  
</group>  
</dataset>
```

# BMS Field Validation Extensions

---

Elastic Transaction Platform provides additional BMS keywords that can be used to provide *client-side* field formatting and validation. This can help to reduce mainframe processing cycles.

## FMASK

---

The FMASK (Fixed Mask) keyword is typically used for formatting input fields such as phone numbers, dates, SSN, credit-cards etc. It can also be used for formatting custom and/or general purpose field types.

### General format

#### Format 1:

**FMASK=(MASK)**

#### Format 2:

**FMASK=(MASK,CUSTOM-MASK)**

#### Format 3:

**FMASK=(MASK,CUSTOM-MASK,PLACEHOLDER)**

### General rules

#### All formats:

1. MASK is a string that may contain mask types. The predefined mask types and their associated regular expressions are:
  - **9**: [0-9] – a numeric character
  - **D**: [0-9\.] – a decimal character
  - **X**: [A-Za-z0-9] – a strictly defined alphanumeric character
  - **A**: [A-Za-z] – an alphabetic character
  - **\***: [.] – any permissible character
  - **B**: [ ] – a blank space character
  - **S**: [+ -] – a sign character
  - **\$**: [\$£€] – a common currency sign character
  - **?**: – any characters following are optional

#### Examples:

1. FMASK=(AAA-999)
  - A field containing 7 characters.

- The first 3 characters entered must be alphabetic.
  - The 4<sup>th</sup> character will automatically be inserted as '-'.
  - The last 3 characters entered must be numeric.
2. FMASK=((999) 999-9999? x99999)
- A field representing a phone number containing 21 characters (the '?' is a function marker, not a character).
  - All characters before '?' are required.
  - All characters after '?' are optional.

**Format 2:**

1. CUSTOM-MASK is a regular expression that provides a custom mask for any position in the MASK string that is marked with a '~'.

**Example:**

1. FMASK=(99~, [A-C])
- A field containing 3 characters.
  - The first 2 characters entered must be numeric.
  - The last character '~' identifies a custom mask is to be applied.
  - The CUSTOM-MASK is defined as the regular expression '[A-C]', which in this example means that the last character entered must be either 'A' or 'B' or 'C'.

**Format 3:**

1. PLACEHOLDER is a string that can be used to indicate to the user the format of the input expected. The PLACEHOLDER is displayed in the input field but does not affect the actual content of the input field.

**Example:**

1. FMASK=(99-99-9999,,MM-DD-YYYY)
- A field representing a date containing 10 characters (including the '-' separator characters).
  - The field will accept only numeric input. The separator characters '-' are inserted automatically.
  - The PLACEHOLDER is defined as the string 'MM-DD-YYYY'.
  - NOTE: The 2<sup>nd</sup> parameter in this example is empty (this is the CUSTOM-MASK parameter and has not been set).

# NMASK

---

The NMASK (Numeric Mask) keyword is typically used for formatting integer & decimal input fields such as general purpose numbers and currency.

Currency number masks automatically include thousand-separators.

## General format

### Format 1:

NMASK=()

### Format 2:

NMASK=(INT | DEC ,NUM | CUR)

### Format 3:

NMASK=(INT | DEC ,NUM | CUR ,MINVALUE ,MAXVALUE)

## General rules

### Format 1:

1. This is equivalent to NMASK=(INT,NUM). See Format 2.

### Format 2:

1. The 1<sup>st</sup> argument can be either INT (integer) or DEC (decimal).
2. The 2<sup>nd</sup> argument can be either NUM (number) or CUR (currency).

### Examples:

1. NMASK=(INT,NUM)
  - All characters entered must be numeric.
  - Input field is defined as an integer number (e.g. 12345), with no decimal component and no thousand-separator.
2. NMASK=(DEC,CUR)
  - All characters entered must be numeric or a decimal point.
  - Input field is defined as decimal currency (e.g. 12,345.00).
  - The ',' thousand-separator is automatically inserted.

### Format 3:

1. MINVALUE & MAXVALUE can be used to establish formatted minimum & maximum values for the mask.

### Example:

1. NMASK=(DEC,CUR,0.0000,999.9999)

- All characters entered must be numeric or a decimal point.
- Input field is defined as decimal currency and will accept any value between 0.0000 and 999.9999
- Format implies a number < 1000 with up to 4 decimals permitted.
- The ‘,’ thousand-separator is automatically inserted.

## EVENT

---

The EVENT keyword is used to perform custom input field validation via JavaScript functions.

### General format

#### Format 1:

**EVENT=(myJavaScriptFunction,[FORM | FIELD],%V MSG,[arg1,arg2,...])**

### General rules

#### Format 1:

1. ‘myJavaScriptFunction’ is the name of the JavaScript function you want to call. Any JavaScript function can be called. By default, ETP provides 3 functions:
  - validateRange – tests if the field input is within a specified range.
  - validateDate – tests if field input is a valid date.
  - validateState – tests if field input is a valid US state.
2. The 2<sup>nd</sup> argument can be either FORM (if testing the submit data from the HTML form) or FIELD (if testing the data input to the HTML form).
3. The 3<sup>rd</sup> argument can be used to pass a custom text message to the function for processing. For example, the text message may be used to display a specific error message in a modal dialog.
4. The %V substitution string in the 3<sup>rd</sup> argument will be replaced by the contents of the input field.
5. Subsequent arguments (i.e. arg1, arg2, ...) are passed through “as is” to the JavaScript function.

#### Examples:

1. EVENT=(validateRange,FORM,%V IS NOT A VALID OPTION. PLEASE ENTER 1 THROUGH 5.,1,5)
  - Upon submission of the FORM, the content (%V) is tested to see if it is within the range of 1 through 5 (the range is specified by the last 2 parameters that are passed to the ‘validateRange’ function).
  - If the range is valid, nothing else will happen and processing will continue.

- If the range is invalid, a modal dialog is displayed with the message passed to the 'validateRange' function, and the cursor is set to the location of the input field so the input can be corrected.
2. EVENT=(validateDate,FIELD,%V IS NOT A VALID DATE. PLEASE RE-ENTER.,-)
- Upon exit of the input FIELD, the contents (%V) are tested to see if they are a valid date.
  - If a separator is being used to format the date, then this is passed as the last argument to the 'validateDate' function.
  - If the date is valid, nothing else will happen and processing will continue.
  - If the date is invalid, a modal dialog is displayed with the message passed to the 'validateDate' function, and the cursor is set to the location of the input field so the input can be corrected.
3. EVENT=(validateState,FIELD,%V IS NOT A VALID STATE. PLEASE RE-ENTER.)
- Upon exit of the input FIELD, the contents (%V) are tested to see if they are a valid US state.
  - If the state is valid, nothing else will happen and processing will continue.
  - If the state is invalid, a modal dialog is displayed with the message passed to the 'validateState' function, and the cursor is set to the location of the input field so the input can be corrected.

## HTMLINPUT

---

The HTMLINPUT keyword is used to inject a string into the input field's HTML INPUT tag. This can be useful for overriding existing attributes or adding others.

### General format

**Format 1:**

**HTMLINPUT=STRING**

### General rules

**Format 1:**

1. STRING must be a valid HTML INPUT attribute.

**Example:**

1. HTMLINPUT=style="text-align: center"
  - This will override any other settings for this attribute.

# ETP Command Syntax

Elastic Transaction Platform service commands supported include all IBM CICS API commands:

Service	Commands
Built-in Function	BIF DEEDIT
Basic Mapping Support	RECEIVE MAP, SEND MAP, SEND TEXT, SEND PAGE, PURGE MESSAGE, ROUTE, RECEIVE PARTN, SEND PARTNSET, SEND CONTROL
Counter	DEFINE, GET, UPDATE, DELETE, REWIND, QUERY COUNTER and DCOUNTER
Date/Time	ASKTIME ABSTIME, FORMATTIME
Document	DOCUMENT CREATE, INSERT, RETRIEVE, SET
Dump Control	DUMP TRANSACTION
Execute Interface	ADDRESS, HANDLE AID, HANDLE CONDITION, ASSIGN, IGNORE CONDITION, PUSH HANDLE, POP HANDLE, ADDRESS SET
File Control targeting SQL/JDBC	READ, WRITE, REWRITE, DELETE, UNLOCK, STARTBR, READNEXT, READPREV, ENDBR, RESETBR
Interval Control	DELAY, POST, START, START ATTACH, START BREXIT, RETRIEVE, CANCEL, WAIT EVENT
Journal	JOURNAL, WRITE JOURNALNUM, WAIT JOURNALNUM, WRITE JOURNALNAME, WAIT JOURNALNAME, WAIT JOURNAL
Operator	WRITE OPERATOR
Program Control	LINK, XCTL, LOAD, RETURN, RELEASE, ABEND, HANDLE ABEND
Recovery	SYNCPOINT, SYNCPOINT ROLLBACK
Security	SIGNON, SIGNOFF, VERIFYIFY PASSWORD, QUERY SECURITY, CHANGE PASSWORD
Spool	SPOOLOPEN, SPOOLREAD, SPOOLWRITE, SPOOLCLOSE
Storage Control	GETMAIN, FREEMAIN
Task Control	SUSPEND, ENQ, DEQ
Task Management	CHANGE TASK
TCP/IP	EXTRACT TCPIP, excluding EXTRACT CERTIFICATE
Temporary Storage	WRITEQ TS, READQ TS, DELETEQ TS
Trace Control	ENTER TRACENUM, ENTER, TRACE, excluding MONITOR
Transient Data Control	WRITEQ TD, READQ TD, DELETEQ TD
Web	WEB RECEIVE, WEB SEND, WEB READ HTTPHEADER, WEB STARTBROWSE HTTPHEADER/FORMFIELD/COOKIE, WEB READ/READNEXT HTTPHEADER/FORMFIELD/COOKIE, WEB ENDBROWSE HTTPHEADER/FORMFIELD/COOKIE, WEB WRITE, WEB EXTRACT, WEB RETRIEVE

## Restrictions

Application Programming Interface (API) commands are supported except for:

- EXTRACT CERTIFICATE (TCP/IP)
- MONITOR (Trace)
- Batch Data Interchange service is not supported
- Distributed Processing service is not supported (APPC, direct terminal control)
- Sub-system capabilities not supported by the ETP sub-system are the System Programming Interface (SPI) commands and conversational transactions.

## ABEND

---

Program controlled user abnormal termination.

### Syntax:

```
ABEND  
    [ABCODE(data-value)]  
    [CANCEL]  
    [NODUMP]
```

### Service:

Program Control

### Description:

An abnormal termination, or abend, propagates from the creation point upwards through the stack until either a programmatic abend handler is found or until the included abend handler is found. The included abend handler creates a program dump so long as NODUMP is not specified.

### Options:

#### **ABCODE(data-value)**

Data-value is the four-character abnormal termination code, visible in the final abend handling dump and available to the abend handler if present.

#### **CANCEL**

Ignore any handlers specified by HANDLE ABEND commands.

#### **NODUMP**

Request no dump be made at final abend handling.

# ADDRESS

---

Establish a pointer to the address of a system area.

## Syntax:

```
ADDRESS
    [ACEE(pointer-ref)]
    [COMMAREA(pointer-ref)]
    [CSA(pointer-ref)]
    [CWA(pointer-ref)]
    [EIB(pointer-ref)]
    [TCTUA(pointer-ref)]
    [TWA(pointer-ref)]
```

## Service:

Execute Interface Program

## Setup:

### **cwa.jndi=cwa\_jndi\_name**

The name of the CWA in JNDI is 'cwa' by default. This allows an alternate JNDI storage name.

### **cwa.value**

This provides an initial value for the CWA if none is currently present. If the CWA is already established, this is ignored.

### **cwa.size**

This provides the size of the CWA in bytes. This defaults to 256.

### **twa.size**

This provides the size of the TWA in bytes. This defaults to 256.

### **eib.size**

This provides the size of the EIB in bytes. This defaults to 256.

### **tctua.size**

This provides the size of the TCTUA in bytes. This defaults to 256.

## Options:

### **ACEE(pointer-ref)**

Establishes linkage to the Access Control Environment Element, information pertinent to the security manager. This may be null.

**COMMAREA(pointer-ref)**

Establishes linkage to the commarea. As the COMMAREA is already passed to the program in the PROCEDURE DIVISION USING or initial linkage, this is useful primarily for access from other called programs where the commarea was not passed. This may be null.

**CSA(pointer-ref)**

As the Common System Area (CSA) has been made obsolete by the ASSIGN command, an abend code of ASRD is thrown if access to the CSA is attempted. Always use ASSIGN for access to information originally available via the CSA.

**CWA(pointer-ref)**

Establishes linkage to the Common Work Area (CWA), user storage common to all tasks within application server. It is stored as a JNDI element named 'cwa'. As a direct pointer may not be shared in an application server environment, a copy is taken when first referenced and the value of the memory is restored to JNDI upon the task's end, if modified. As even with direct pointers there would be race conditions, generally do not modify the CWA on other than an occasional basis (such as once per day for a custom date format). This may be null.

**EIB(pointer-ref)**

Establishes linkage to the Execute Interface Block (EIB). As the EIB is already passed to the program in the PROCEDURE DIVISION USING or initial linkage, this is useful primarily for access from other called programs where the EIB was not passed.

Be aware that the method used for handling the EIB means that a variable by the correct name must be present in the program in order to be set by the EXEC command. When called directly from Java, the EIB value must be passed as a parameter; the EIB block itself is not used.

**TCTUA(pointer-ref)**

Establishes linkage to the Terminal Control Table User Area, storage unique to the virtual terminal session. This has little use in command-level programming.

**TWA(pointer-ref)**

Establishes linkage to the Transaction Work Area (TWA), storage unique to the current task. This has little use in command-level programming.

**Conditions:**

None

# ADDRESS SET

---

Set the address of a pointer.

This command is generally not used in new code, as the COBOL SET verb may be used directly.

## Syntax:

```
ADDRESS
    [SET(data-area) USING(pointer-ref)]
    [SET(pointer-ref) USING(data-area)]
```

## Service:

Execute Interface Program

## Setup:

None

## Options:

### **SET(data-area)**

Specifies that the data-area's pointer is modified to be pointing to the using parameter.

### **SET(pointer-ref)**

Specifies that the pointer is modified to be pointing to the using parameter.

### **USING(pointer-ref)**

Specifies the pointer source to be used by the SET option.

### **USING(data-area)**

Specifies the data-area source's pointer to be used by the SET option.

## Conditions:

None

# ASKTIME

---

Retrieve the current timestamp. The FORMATTIME command may be used to transform the returned timestamp to human readable formats.

ASKTIME also sets EIBDATE and EIBTIME, if present in the calling program.

### Syntax:

```
ASKTIME
      [ABSTIME(data-area)]
      [MILLISECONDS]
```

### Service:

Date Time

### Setup:

None

### Options:

#### **ABSTIME**

Retrieves current date/time in milliseconds since January 1, 1900 at 00:00.

#### **MILLISECONDS**

Do not artificially decrease accuracy to hundredths of a second for compatibility.

### Conditions:

None

## ASSIGN

---

Assign the contents of a transaction platform value to a given data-area. This is used for retrieving internal information safely. All options are for returning data.

### Syntax:

```
ASSIGN
      [ABCODE(data-area)] [ABDUMP(data-area)]
      [ABPROGRAM(data-area)] [ACTIVITY(data-area)]
      [ACTIVITYID(data-area)] [ALTSCRNHT(data-area)]
      [ALTSCRNWD(data-area)] [APLKYBD(data-area)]
```

[APLTEXT(data-area)] [APPLID(data-area)]  
[ASRAINTRPT(data-area)] [ASRAKEY(cvda)]  
[ASRAPSW(data-area)] [ASRAREG(data-area)]  
[ASRASPC(cvda)] [ASRASTG(cvda)]  
[BRIDGE(data-area)] [BTRANS(data-area)]  
[CMDSEC(data-area)] [COLOR(data-area)]  
[CWALENG(data-area)] [DEFSCRNHT(data-area)]  
[DEFSCRNWD(data-area)] [DELIMITER(data-area)]  
[DESTCOUNT(data-area)] [DESTID(data-area)]  
[DESTIDLENG(data-area)] [DSSCS(data-area)]  
[DS3270(data-area)] [EWASUPP(data-area)]  
[EXTDS(data-area)] [FACILITY(data-area)]  
[FCI(data-area)] [GCHARS(data-area)]  
[GCODES(data-area)] [GMMI(data-area)]  
[HIGHLIGHT(data-area)] [INITPARM(data-area)]  
[INITPARMLEN(data-area)] [INPARTN(data-area)]  
[INVOKINGPROG(data-area)] [KATAKANA(data-area)]  
[LANGINUSE(data-area)] [LDCMNEM(data-area)]  
[LDCNUM(data-area)] [MAPCOLUMN(data-area)]  
[MAPHEIGHT(data-area)] [MAPLINE(data-area)]  
[MAPWIDTH(data-area)] [MSRCONTROL(data-area)]  
[NATLANGINUSE(data-area)] [NETNAME(data-area)]  
[NEXTTRANSID(data-area)] [NUMTAB(data-area)]  
[OPCLASS(data-area)] [OPERKEYS(data-area)]  
[OPID(data-area)] [OPSECURITY(data-area)]  
[ORGABCODE(data-area)] [OUTLINE(data-area)]  
[PAGENUM(data-area)] [PARTNPAGE(data-area)]  
[PARTNS(data-area)] [PARTNSET(data-area)]  
[PRINSYSID(data-area)] [PROCESSS(data-area)]  
[PROCESSTYPE(data-area)] [PROGRAM(data-area)]  
[PS(data-area)] [QNAME(data-area)]  
[RESSEC(data-area)] [RESTART(data-area)]  
[RETURNPROG(data-area)] [SCRNHT(data-area)]  
[SCRNWD(data-area)] [SIGDATA(data-area)]  
[SOSI(data-area)] [SPJAVAVERSION(data-area)]  
[SPJAVAVENDOR(data-area)] [SPJAVAVENDORURL(data-area)]  
[SPJAVAHOME(data-area)] [SPJAVACLASSVERSION(data-area)]

[SPJAVACLASSPATH(data-area)] [SPOSNAME(data-area)]  
[SPOSARCH(data-area)][SPOSVERSION(data-area)]  
[SPFILESEPARATOR(data-area)] [SPPATHSEPARATOR(data-area)]  
[SPLINESEPARATOR(data-area)] [STARTCODE(data-area)]  
[STARTIONID(data-area)] [SYSID(data-area)]  
[TASKPRIORITY(data-area)] [TCTUALENG(data-area)]  
[TELLERID(data-area)] [TERMCODE(data-area)]  
[TERMPRIORITY(data-area)] [TEXTKYBD(data-area)]  
[TEXTPRINT(data-area)] [TRANPRIORITY(data-area)]  
[TWALENG(data-area)] [UNATTEND(data-area)]  
[USERID(data-area)] [USERNAME(data-area)]  
[USERPRIORITY(data-area)] [VALIDATION(data-area)]  
[VERSIONMAJOR(data-area)] [VERSIONMINOR(data-area)]  
[VERSIONSERVICE(data-area)] [VERSION(data-area)]

### Service:

Execute Interface Program

### Setup:

None

### Options:

#### **ABCODE(data-area)**

4-character current abend code, blanks if no abend.

#### **ABDUMP(data-area)**

X"FF" if dump has been produced, X"00" otherwise.

#### **ABPROGRAM(data-area)**

8-character program name of abending program.

#### **ACTIVITY(data-area)**

BTS only, until Business Transaction Services supported throws condition INVREQ.

#### **ACTIVITYID(data-area)**

BTS only, until Business Transaction Services supported throws condition INVREQ.

#### **ALTSCRNHT(data-area)**

Alternate screen height for terminal.

**ALTSCRNWD(data-area)**

Alternate screen width for terminal.

**APLKYBD(data-area)**

X"FF" if APL keyboard feature, X"00" otherwise.

**APLTEXT(data-area)**

X"FF" if terminal keyboard has APL text feature, X"00" otherwise.

**APPLID(data-area)**

8-character applid of system owning transaction.

**ASRAINTRPT(data-area)**

Not supported. If supported, would return 8-character data-area containing ILC and PIC of last abend of type ASRA, ASRB, ASRD or AICA.

**ASRAKEY(cvda)**

Returns the execution key of the previous ASRA-style interrupt. This could return any of the following cvda values:

CICSEXECKEY

USEREXECKEY

NONCICS

NOTAPPLIC

The current release always returns NOTAPPLIC.

**ASRAPSW(data-area)**

8-character Program Status Word of last ASRA-style interrupt. The current release always returns binary zeroes.

**ASRAREG(data-area)**

64-byte data area holding general registers 0...15 of last AREA-style interrupt. The current release always returns binary zeroes, as this register set is mainframe specific.

**ASRASPC(cvda)**

Returns the space in effect at the time of the previous ASRA-style interrupt. This could return any of the following cvda values:

SUBSPACE

BASESPACE

NOTAPPLIC

The current release always returns NOTAPPLIC.

**ASRASTG(cvda)**

Returns the storage addressed at the time of the previous ASRA-style interrupt. This could return any of the following cvda values:

CICS  
USER  
READONLY  
NOTAPPLIC

The current release always returns NOTAPPLIC.

**BRIDGE(data-area)**

4-character TRANSID of bridge monitor transaction that issued the START BEXIT TRANSID. command. Blanks are returned if this is not applicable.

**BTRANS(data-area)**

X"FF" if terminal support background transparency, X"00" otherwise.

**CMDSEC(data-area)**

X"FF" if command security defined for task, X"00" otherwise.

**COLOR(data-area)**

X"FF" if terminal supports extended color capability, X"00" otherwise.

**CWALENG(data-area)**

Length of the Common Work Area (CWA).

**DEFSCRNHT(data-area)**

Default screen height for terminal.

**DEFSCRNWD(data-area)**

Default screen width for terminal.

**DELIMITER(data-area)**

1-byte data-link control character for 3600. This will be one of:

X"80"	end-of-text
X"40"	end-of-block
X"20"	inter-record separator
X"10"	start of header
X"08"	transparent input

**DESTCOUNT(data-area)**

Following BMS ROUTE, number of different terminal types in route list. Within BMS overflow, relative overflow control number.

**DESTID(data-area)**

BDI only, until Batch Data Interchange is supported throws the condition INVREQ. 8-character outboard destination.

**DESTIDLENG(data-area)**

BDI only, until Batch Data Interchange is supported throws the condition INVREQ. Length of destination identifier in DESTID.

**DSSCS(data-area)**

X"FF" if facility is basic SCS data stream device, X"00" otherwise.

**DS3270(data-area)**

X"FF" if 3270 data stream device, X"00" otherwise.

**EWASUPP(data-area)**

X"FF" if Erase Write Alternative supported, X"00" otherwise.

**EXTDS(data-area)**

X"FF" if terminal supports 3270 extended data stream, X"00" otherwise.

**FACILITY(data-area)**

4-byte principal facility identifier.

**FCI(data-area)**

1-byte Facility Control Indicator.

**GCHARS(data-area)**

Graphic character set global identifier (GCSGID), 1...65,534.

**GCODES(data-area)**

Code page global identifier (CPGID), 1...65,534.

**GMMI(data-area)**

X"FF" if good morning message for terminal, X"00" otherwise.

**HILIGHT(data-area)**

X"FF" if terminal supports extended highlight capability, X"00" otherwise.

**INITPARM(data-area)**

60-character initialization parameter. If none, binary zeroes.

The setup for the initparm is:

```
program.<program_name>.initparm=<initparm_value>
```

**INITPARMLEN(data-area)**

Length of INITPARM. If no INITPARM, then zero.

**INPARTN(data-area)**

1- or 2-character name of most recent input partition.

**INVOKINGPROG(data-area)**

8-character name of program that used LINK or XCTL to transfer control to current program. Blanks if at highest level.

**KATAKANA(data-area)**

X"FF" if facility supports katakana, X"00" otherwise.

**LANGINUSE(data-area)**

3-byte mnemonic code for language in use, corresponds to NATLANGINUSE.

**LDCMNEM(data-area)**

2-byte LDC (Logical Device Code) mnemonic of destination that overflowed.

**LDCNUM(data-area)**

1-byte LDC numeric value of destination that overflowed.

**MAPCOLUMN(data-area)**

Origin column of most recently positioned map.

**MAPHEIGHT(data-area)**

Height of most recently positioned map.

**MAPLINE(data-area)**

Origin line of most recently positioned map.

**MAPWIDTH(data-area)**

Width of most recently positioned map.

**MSRCONTROL(data-area)**

X"FF" if terminal supports MSR (Magnetic Slot Reader), X"00" otherwise.

**NATLANGINUSE(data-area)**

1-byte mnemonic code for language in use, corresponds to LANGINUSE.

**NETNAME(data-area)**

8-character name of logical unit in VTAM network, if applicable.

**NEXTTRANSID(data-area)**

4-character name of next transaction, if available.

**NUMTAB(data-area)**

1-byte number of tabs required to position print element for 2980.

**OPCLASS(data-area)**

24-bits, BMS operator class for routing terminal messages.

**OPERKEYS(data-area)**

Supported for compatibility only, 8 null bytes.

**OPID(data-area)**

3-character operator ID.

**OPSECURITY(data-area)**

3-bytes of zeroes, this field is not used in current systems.

**ORGABCODE(data-area)**

4-byte original abend code.

**OUTLINE(data-area)**

X"FF" if terminal supports field outlining, X"00" otherwise.

**PAGENUM(data-area)**

Page number for overflow.

**PARTNPAGE(data-area)**

2-byte name of partition having caused page overflow.

**PARTNS(data-area)**

X"FF" if terminal supports partitions, X"00" otherwise.

**PARTNSET(data-area)**

1...6 character application partition set.

**PRINSYSID(data-area)**

4-character name by which the other system is known locally.

**PROCESSS(data-area)**

BTS only, until Business Transaction Services supported throws condition INVREQ.

**PROCESSTYPE(data-area)**

BTS only, until Business Transaction Services supported throws condition INVREQ.

**PROGRAM(data-area)**

8-character program name of the current program.

**PS(data-area)**

X"FF" if terminal has programmed symbols capability, X"00" otherwise.

**QNAME(data-area)**

4-character queue name that initiated the task, if task initiated by ATI.

**RESSEC(data-area)**

"X" if resource security checking in effect, blank otherwise.

**RESTART(data-area)**

X"FF" if restart of task has occurred, X"00" otherwise.

**RETURNPROG(data-area)**

8-character program name of the program to which the current program will return.

**SCRNHT(data-area)**

Height of 3270 screen.

**SCRNWD(data-area)**

Width of 3270 screen.

**SIGDATA(data-area)**

4-byte inbound signal.

**SOSI(data-area)**

X"FF" is mixed DBCS is supported, X"00" otherwise.

**SPJAVAVERSION(data-area)**

System Property, the Java version.

**SPJAVAVENDOR(data-area)**

System Property, the Java vendor.

**SPJAVAVENDORURL(data-area)**

System Property, the Java vendor URL.

**SPJAVAHOME(data-area)**

System Property, the Java home.

**SPJAVACLASSVERSION(data-area)**

System Property, the Java class version.

**SPJAVACLASSPATH(data-area)**

System Property, the Java classpath.

**SPOSNAME(data-area)**

System Property, the Operating System name.

**SPOSARCH(data-area)**

System Property, the Operating System architecture.

**SPOSVERSION(data-area)**

System Property, the Operating System version.

**SPFILESEPARATOR(data-area)**

System Property, the file separator character, as in '/' or '\'.

**SPPATHSEPARATOR(data-area)**

System Property, the path separator character, as in ':' or ';'.

**SPLINESEPARATOR(data-area)**

System Property, the line separator, as in 13 10.

**STARTCODE(data-area)**

2-byte indicator specifying how task was started.

D	DPL without SYNCONRETURN.
DS	DPL with SYNCONRETURN.

QD	Transient data queue.
S	START without data.
SD	START with data.
SZ	FEPI START.
TD	Terminal.
U	User-attached.

**STARTIONID(data-area)**

1-byte station ID for 2980 terminal.

**SYSID(data-area)**

4 character system ID.

**TASKPRIORITY(data-area)**

Task priority, 0...255.

**TCTUALENG(data-area)**

Length of Terminal Control Table User Area (TCTUA).

**TELLERID(data-area)**

1-byte teller ID for 2980 terminal.

**TERMCODE(data-area)**

2-byte terminal code for terminal associated to task. The first byte is terminal type, the second byte is model number.

**TERMPRIORITY(data-area)**

Terminal priority, 0...255. If separate priorities are not supported, 0 is returned.

**TEXTKYBD(data-area)**

X"FF" is terminal supports TEXTKYBD, X"00" otherwise.

**TEXTPRINT(data-area)**

X"FF" if terminal supports TEXTPRINT, X"00" otherwise.

**TRANPRIORITY(data-area)**

Transaction priority, 0...255. If separate priorities are not supported, 0 is returned.

**TWALENG(data-area)**

Length of Transaction Work Area (TWA).

**UNATTEND(data-area)**

X"FF" is terminal is in unattended mode, X"00" otherwise.

**USERID(data-area)**

ID of the user. Traditionally up to 8 characters, but it may be more depending upon user id's allocated by the system. The assign will truncate if necessary.

**USERNAME(data-area)**

Name of the user. Traditionally up to 20 characters, but it may be more depending upon user names allocated by system. The assign will truncate if necessary.

**USERPRIORITY(data-area)**

Operator priority, 0...255. If separate priorities are not supported, 0 is returned.

**VALIDATION(data-area)**

X"FF" is terminal supports validation, X"00" otherwise.

**VERSIONMAJOR(data-area)**

Returns the major release version of the transaction platform.

**VERSIONMINOR(data-area)**

Returns the minor release version of the transaction platform.

**VERSIONSERVICE(data-area)**

Returns the service release version of the transaction platform.

**VERSION(data-area)**

Returns a text representation of the version number, as major.minor.service.

**Conditions:**

INVREQ	RESP2
1	No signed in user present.
2	No BMS command in use.
3	No BDI, Batch Data Interchange in use.
4	The task was not started using ATI, Automatic Transaction Initiation.
5	The task has no facility.
6	Not operating under BTS, Business Transaction Services.
200	Not valid in DPL, Distributed Program Link.
NOTAUT H	The requested assignment has not been authorized, generally by the application server itself.

## **BIF DEEDIT**

---

Format alphanumeric data into numeric data. This function is obsolete.

It retains all numeric characters (digits 0 through 9), and right-aligns them with zero-fill on the left. It does not preserve the decimal point or signed, zoned digits.

### Syntax:

BIF DEEDIT

### Service:

Built In Function

### Setup:

None

### Options:

#### FIELD

The field is the data that is the alphanumeric source and numeric destination.

#### LENGTH

The length is a binary number describing the length of the field. If not given, the length of the FIELD variable is used.

### Conditions:

None

### Example:

```
01 bif-1 pic x(17).  
  
move "ab1cd2ef3gh4ij5kl" to bif-1  
  
exec transaction  
  web send text("BIF FROM: " bif-1)  
end-exec  
  
exec transaction  
  bif deedit field(bif-1)  
end-exec  
  
exec transaction  
  web send text("BIF TO: " bif-1)  
end-exec
```

### Output:

```
BIF FROM:  ab1cd2ef3gh4ij5kl  
BIF TO:    00000000000012345
```

# CANCEL

---

Cancel a previous interval control request. The request must be still waiting to be performed in order to be successfully cancelled.

## Syntax:

```
CANCEL
    [REQID(data-value)
    [SYSID(data-value)]
    [TRANSID(data-value)]
    ]
```

## Service:

Interval Control

## Setup:

None

## Options:

### **REQID(data-value)**

Specifies request ID (1 to 8 characters) used with START, START ATTACH, POST or DELAY command.

### **SYSID(data-value)**

Specifies the system ID of the request to cancel, if used.

### **TRANSID(data-value)**

Specifies the transaction ID of the request to cancel, if used.

## Conditions:

NOTFND	The REQID was not found.
--------	--------------------------

# CHANGE PASSWORD

---

This command is dependent upon the capabilities of the External Security Manager.

### Syntax:

```
CHANGE PASSWORD(data-value)
      NEWPASSWORD(data-value)
      USERID(data-value)
      [ESMREASON(data-area)]
      [ESMRESP(data-area)]
```

### Service:

Security

### Setup:

None

### Options:

#### **ESMREASON(data-area)**

Retrieves the external security manager's reason code.

#### **ESMRESP(data-area)**

Retrieves the external security manager's response code.

#### **NEWPASSWORD(data-value)**

Specifies the new password, up to 8 characters.

#### **PASSWORD(data-value)**

Specifies the current password, up to 8 characters.

#### **USERID(data-value)**

Specifies the user id, up to 8 characters.

### Conditions:

INVREQ	Invalid request.
NOTAUTH	Not authorized.
USERIDERR	The user id is unknown

## CHANGE TASK

---

Change a task's operating priority, yielding the CPU in the process.

**Syntax:**

```
CHANGE TASK
      [PRIORITY(data-value)]
```

**Service:**

Task Management

**Setup:**

None

**Options:****PRIORITY**

-1 through 255.

-1 is a no operation, not yielding the CPU.

0 through 255 are priorities. These priorities are scaled to the JVM's levels of priority, which are determined by Thread.MIN\_PRIORITY through Thread.MAX\_PRIORITY, which is 1 through 10 at the time of this writing.

**Conditions:**

INVREQ	RESP2
1	The PRIORITY is not within -1 through 255.
100	The PRIORITY option is not supported within the current environment.

## DEFINE COUNTER

---

Define a counter in the counter service. A counter must be defined before its first use. All counters are referenced by name.

Counters are segregated by pools, with each pool having an independent version of any named counters. The pool is selected by the program code.

**Syntax:**

```
DEFINE { COUNTER(name) | DCOUNTER(name) } [POOL(name)]
      [VALUE(data-value)]
      [MINIMUM(data-value)]
      [MAXIMUM(data-value)]
```

## Service:

Counter

## Setup:

### **pool.name=redirected\_pool\_name**

If not specified, the pool name is the given name.

### **counter.jndi=counter\_jndi\_name**

Specify the JNDI name of the external Counter service. The default is 'CounterService'.

The counter dispatches most of its work off to an external service so that the counters will be shared between tasks.

## Options:

### **COUNTER(name)**

Specifies the name of the 32-bit signed counter.

### **DCOUNTER(name)**

Specifies the name of the 64-bit unsigned counter.

### **POOL(name)**

Specifies the name of the pool of counters. Each Counter service maintains its own pools of counters.

### **VALUE(data-value)**

Sets the initial value of the counter.

### **MINIMUM(data-value)**

Specifies the minimum value for the counter.

### **MAXIMUM(data-value)**

Specifies the maximum value for the counter.

## Conditions:

INVREQ	RESP2
202	Counter name already exists, cannot create duplicate counter.
301	Server error.
302	No space in the pool to create the counter.
303	Unexpected error, such as connectivity loss.
304	Invalid pool.
305	Cannot connect to server.

306	Server abend.
308	This condition cannot occur (options table not loadable).
309	This condition cannot occur (options table error).
310	This condition cannot occur (options user exit).
403	The pool name contains invalid characters.
404	The counter name contains invalid characters.
406	The increment is invalid; it cannot be larger than the total range of the counter.
407	The minimum or maximum value is invalid; negative, or maximum less than minimum

## DELAY

---

Delay the task for a period of time in a CPU-efficient manner.

As delay loops may be optimized away, and the CPU should be free to perform other activities during processing, a delay may be introduced using this command.

INTERVAL or FOR specify a period of time, whereas TIME or UNTIL specify the final time itself.

Note that most foreground sessions contain timeouts and will be terminated if delayed too far.

### Syntax:

```

DELAY
    [REQID(data-value)]
        [INTERVAL(hhmmss)]
    [FOR
        [HOURS(data-value)]
        [MINUTES(data-value)]
    [    SECONDS(data-value)]
    ]
    [TIME(hhmmss)]
    [UNTIL
        [HOURS(data-value)]
        [MINUTES(data-value)]
        [SECONDS(data-value)]
    ]
  ]

```

## Service:

Interval Control

## Setup:

None

## Options:

### **REQID(data-value)**

Specifies a request ID through which this command may be cancelled using the CANCEL command. The request ID is valid only within the same session.

### **INTERVAL(hhmmss)**

Specifies an interval of time duration data-value. The format is a single number with two-digits each hour, minute, second.

### **TIME(hhmmss)**

Specifies a final time data-value. The format is a single number with two-digits each hour, minute, second.

### **FOR**

FOR specifies that HOURS, MINUTES and SECONDS refers to an interval duration.

### **UNTIL**

UNTIL specifies that HOURS, MINUTES and SECONDS refers to a final time.

### **HOURS(data-value)**

Specifies the number of hours, and it must be within 0..99.

### **MINUTES(data-value)**

Specifies the number of minutes, and it must be within 0..59 if other HOURS or **SECONDS is specified.**

### **SECONDS(data-value)**

Specifies the number of seconds, and it must be within 0..59 if other HOURS or **MINUTES is specified.**

## Conditions:

EXPIRED	The interval has already expired.
INVREQ	RESP2

	4 Hours are invalid 5 Minutes are invalid. 6 Seconds are invalid
--	--

## DELETE

---

Delete a record from a given file, interpreted as deleting a row from a database.

### Syntax:

```
DELETE FILE(data-area)
  [
  TOKEN(data-area) |
  RIDFLD(data-area) [KEYLENGTH(data-value)
  [GENERIC [NUMREC(data-area)]]]
  ]
  [SYSID(data-area)]
  [NOSUSPEND]
  [RBA | RRN]
```

### Service:

File Control

### Setup:

**file.<filename>=jdbc:<  
xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**

The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

## Options:

### **FILE(data-area)**

Specifies the name of the dataset, which must be included in setup as a reference to jdbc:filename.xml.

### **TOKEN(data-area)**

Specifies token used by READ UPDATE in order to associate this delete with a prior read.

### **RIDFLD(data-area)**

Specifies the record ID field, the key field.

### **KEYLENGTH(data-value)**

Specifies the length of the key.

### **GENERIC**

The key is generic, specified to the length of keylength.

### **NUMREC(data-area)**

Returns the number of deleted records.

### **SYSID(data-area)**

Specifies the system ID.

### **NOSUSPEND**

Specifies that no waiting is to occur if records are locked.

### **RBA**

Use RBA as the key field name.

### **RRN**

Use RRN as the key field name.

## Conditions:

FILENOTFOUN D	The file was not found.
INVREQ	An invalid request was issued.
IOERR	An input/output file error occurred during the operation
NOTFND	The operation could not find the record.

## DELETE COUNTER

---

Delete the named counter.

## Syntax:

```
DELETE { COUNTER(name) | DCOUNTER(name) } [POOL(name)]
```

## Service:

Counter

## Setup:

**pool.name=redirected\_pool\_name**

If not specified, the pool name is the given name.

**counter.jndi=counter\_jndi\_name**

Specify the JNDI name of the Counter service. The default is 'CounterService'.

## Options:

**COUNTER(name)**

Specifies the name of the 32-bit counter.

**DCOUNTER(name)**

Specifies the name of the 64-bit counter.

**POOL(name)**

Specifies the name of the pool of counters. Each Counter service maintains its own pools of counters.

## Conditions:

INVREQ	RESP2
201	Named counter invalid.
301	Server error.
303	Unexpected error, such as connectivity loss.
304	Invalid pool.
305	Cannot connect to server.
306	Server abend.
308	This condition cannot occur (options table not loadable).
309	This condition cannot occur (options table error).
310	This condition cannot occur (options user exit).
403	The pool name contains invalid characters.

## DELETEQ TD

---

Delete all data within a transient data queue.

### Syntax:

```
DELETEQ TD QUEUE(name)
        [SYSID(name)]
```

### Service:

Transient Data Control

### Setup:

```
queue[.sysid_value].name=uri
[sysid.name=sysid_value]
```

### Options

#### QUEUE(name)

Specifies the name of the transient data queue. The queue must be defined.

#### SYSID(name)

Specifies the system ID. This is used to determine which queue definition is used.

### Conditions

SYSIDERR	An unknown sysid is being used.
QIDERR	1 There is no such queue definition. 2 The queue could not be obtained
INVREQ	The queue could not be deleted. For instance, JMS queues (being externally available) cannot be deleted.

## DELETEQ TS

---

Delete all data within a temporary storage queue.

### Syntax:

```
DELETEQ TS {QUEUE(name) | QNAME(name)}
```

[SYSID(name)]

### Service:

Temporary Storage Control

### Setup:

queue[.sysid\_value].name[.main | .auxiliary]=uri  
[sysid.name=sysid\_value]

### Options

#### QUEUE(name)

Specifies the name of the transient data queue. The queue must be defined.

#### QNAME(name)

See QUEUE.

#### SYSID(name)

Specifies the system ID. This is used to determine which queue definition is used.

### Conditions

SYSIDERR		An unknown sysid is being used
QIDERR	1	There is no such queue definition.
	2	The queue could not be obtained.

## DOCUMENT CREATE

---

Create a new document.

The document is stored in an internal format, manipulated with DOCUMENT commands, and either sent to the web using WEB SEND or retrieved in final form using DOCUMENT RETRIEVE.

### Syntax:

```
DOCUMENT CREATE DOCTOKEN(data-area)  
{{ TEXT(data-area) | BINARY(data-area) | FROM(data-area)} [LENGTH(data-  
value)]} |  
TEMPLATE(name)
```

FROMDOC(data-area)  
[DOCSIZE(data-area)]  
[HOSTCODEPAGE(name)]  
[SYMBOLLIST(data-area) [LISTLENGTH(data-value)]]

### Service:

Document

### Setup:

**template.name=resource**

### Options:

#### **DOCTOKEN(data-area)**

Retrieves a 16-byte document token. The document token must then be passed to other DOCUMENT commands to refer to the same document.

#### **TEXT(data-area)**

Specifies text to be inserted.

#### **BINARY(data-area)**

Specifies binary data to be inserted.

#### **FROM(data-area)**

Specifies a binary image of a previously retrieved document or template.

#### **LENGTH(data-value)**

Specifies the length of the TEXT, BINARY or FROM data.

#### **TEMPLATE(name)**

Specifies a template. The template must be defined.

#### **FROMDOC(data-area)**

Specifies that contents of another document are to be inserted. The data-area is the 16-byte retrieved DOCTOKEN.

#### **DOCSIZE(data-area)**

Retrieves the estimated byte size of the document.

#### **HOSTCODEPAGE(name)**

Specifies the host's code page.

#### **SYMBOLLIST(data-area)**

Specifies a list of symbols and their values in URL-encoded form. Each list agreement is a name, an equals sign (=), and a value. Each list element is separated from each other list element by an ampersand (&).

The value must not contain a plain ampersand, as it is used for element separation, and it must not contain a plain percentage character (%), as it is used for embedding values. The % sign must be followed by two hexadecimal characters, which form a single a character representing the given ASCII value. So, %26 may be used to embed an ampersand (&), and %25 may be used to embed a percentage character (%). A plus sign represents a space, so a plus sign must be embedded as %2B. All other characters are themselves.

Example:

```
SYMBOLLIST('first=George&last=Washington&fullname=George+Washington').
LISTLENGTH(data-value)
```

Specifies the length of the SYMBOLLIST.

### Conditions:

INVREQ	RESP2
1	The document being received FROM is in an invalid format.

NOTFND	RESP2
2	The FROMDOC is not found.
3	The TEMPLATE is not found.
7	The HOSTCODEPAGE is not found.
SYMBOLERR	The symbol name is invalid.
TEMPLATERR	The template is invalid.

## DEQ

---

Release a resource from exclusive use that had been previously allocated by ENQ.

This command is cooperative, so any other tasks wanting the same resource must also allocate it for exclusive use using the same mechanism.

The default implementation uses JNDI. A resource is locked to a single task by binding an object to the JNDI name 'Its\_resource\_<resourcename>', where <resourcename> is the name of the resource to its given length. The resource is unlocked by unbinding the object. (Nesting the enq/deq is left to the individual task when locked between tasks.) The scope of the resource allocation is therefore the scope of JNDI in use, either a single application server or clustered application servers using clustered JNDI.

### Syntax:

```
DEQ RESOURCE(data-area) [LENGTH(data-value)]
```

[UOW | TASK | MAXLIFETIME(cvda)]

**Service:**

Task Control

**Options:**

**RESOURCE(data-value)**

Specifies the name of the resource as a data-area. This is always the content of the specified resource, not the address of the resource. LENGTH should always be specified, but the implicit length of the resource is used if LENGTH is left unspecified.

**LENGTH(data-value)**

Specifies the length of the resource name. If left unspecified, an implicit length of the resource data-area is used, not the address of the resource. This must be between 1 and 255. It must be same as when allocated by ENQ, or the resource will not have the same resource name as intended.

**UOW**

Specifies that the resource is to live until the end of the unit of work. This is the default.

**TASK**

Specifies that the resource is to live until the end of the task.

**MAXLIFETIME(cvda)**

Specifies the maximum lifetime of the resource, either the CVDA for UOW or TASK. The CVDA LUW is accepted as a synonym for UOW. Specify the lifetime in this way if not constant, otherwise use the UOW or TASK tag directly.

**Conditions:**

INVREQ	
2	MAXLIFETIME was set to an invalid CVDA.

LENGERR	
1	LENGTH was not between 1 and 255.

## DOCUMENT INSERT

---

Insert document elements into a document.

## Syntax:

```
DOCUMENT INSERT DOCTOKEN(data-area)
{{ TEXT(data-area) | BINARY(data-area) | FROM(data-area)} [LENGTH(data-
value)]} |
    SYMBOL(name)
    TEMPLATE(name)
    FROMDOC(data-area)
    BOOKMARK(name)
    [DOCSIZE(data-area)]
    [HOSTCODEPAGE(name)]
    [AT(name)]
    [TO(name)]
```

## Service:

Document

## Setup:

**template.name=resource**

## Options:

### **DOCTOKEN(data-area)**

Specifies the document into which the element is inserted; this is obtained originally from a DOCUMENT CREATE.

### **TEXT(data-area)**

Specifies text to be inserted.

### **BINARY(data-area)**

Specifies binary data to be inserted.

### **FROM(data-area)**

Specifies a binary image of a previously retrieved document or template.

### **LENGTH(data-value)**

Specifies the length of the TEXT, BINARY or FROM data.

### **SYMBOL(name)**

Specifies a symbol. The symbol's current content value is inserted, not the symbol itself.

### **TEMPLATE(name)**

Specifies a template. The template must be defined.

**FROMDOC(data-area)**

Specifies that contents of another document are to be inserted. The data-area is the 16-byte retrieved DOCTOKEN.

**BOOKMARK(name)**

Specifies that a bookmark is to be inserted. The bookmark may be used to note the current location for later inserts.

**DOCSIZE(data-area)**

Retrieves the estimated byte size of the document.

**HOSTCODEPAGE(name)**

Specifies the host's code page.

**AT(name)**

If TO is not specified, then the insertion occurs immediately after the bookmark name.

If TO is specified, then the contents between AT and TO bookmarks are removed, being replaced by this insertion element.

**TO(name)**

If AT is not specified, then the insertion occurs immediately before the bookmark name.

If AT is specified, then the contents between AT and TO bookmarks are removed, being replaced by this insertion element.

**Conditions:**

DUPREC	A duplicate bookmark is being defined.
--------	--

INVREQ	RESP2
0	The bookmark TO is incorrectly before the bookmark AT.
1	The document being received FROM is in an invalid format.
2	The bookmark is invalid.

NOTFND	RESP2
1	The DOCUMENT is not found
2	The FROMDOC is not found.
3	The TEMPLATE is not found
4	The SYMBOL is not found.
5	The AT bookmark is not found.
6	The TO bookmark is not found.
7	The HOSTCODEPAGE is not found.

TEMPLATER R	The template is invalid.
----------------	--------------------------

## DOCUMENT RETRIEVE

---

Retrieve a copy of a document.

### Syntax:

```
DOCUMENT RETRIEVE DOCTOKEN(data-area)
                    INTO(data-area)
                    LENGTH(data-area)
                    MAXLENGTH(data-value)
                    CLNTCODEPAGE(name)
                    DATAONLY
```

### Service:

Document

### Setup:

None

### Options:

#### **DOCTOKEN(data-area)**

Specifies the document token, the document to retrieve.

#### **INTO(data-area)**

Returns the data itself.

#### **LENGTH(data-area)**

Returns the length of the data.

#### **MAXLENGTH(data-value)**

Specifies the maximum length of data expected.

#### **CLNTCODEPAGE(name)**

Specifies the client code page.

#### **DATAONLY**

Return only data, not meta-data such as internal bookmarks.

### Conditions:

LENGERR	RESP2
1	MAXLENGTH is less than or equal to zero.
2	The document is truncated because the buffer is too small.

NOTFND	RESP2
1	The document is not found.
7	The client codepage is invalid.

## DOCUMENT SET

---

Add or set symbols within the specified document.

### Syntax:

```
DOCUMENT SET
    DOCTOKEN(data-area)
    {
    { SYMBOL(name) [ NAMELENGTH(data-value) ] VALUE(data-area) } |
    { SYMBOLLIST(data-area) }
    }
    LENGTH(data-value)
```

### Service:

Document

### Options:

#### **DOCTOKEN(data-area)**

Specifies a document token, created using earlier document commands.

#### **SYMBOL(name)**

Specifies a symbol name. The name is whitespace trimmed when used.

#### **NAMELENGTH(data-value)**

Specifies the length of the symbol name. (The default is the whitespace trimmed length.)

#### **SYMBOLLIST(data-area)**

Specifies a list of symbols. The list is of the form name=value, separated by semicolons (;). The values are restricted, such that a percent sign followed

by two hexadecimal digits is an ASCII character, ampersand is not allowed (replaced by %26), and pluses (+) represent spaces (replaced by %2B).

**LENGTH(data-value)**

Specifies the length of the symbol value or of the symbol list.

**VALUE(data-area)**

Specifies the value of the symbol.

**Conditions:**

NOTFND	RESP2
1	The document does not exist.

SYMBOLERR	RESP2
0	Neither SYMBOL nor SYMBOLLIST was specified, or if specified were invalid.

## DUMP TRANSACTION

---

Dump information about the transaction programmatically. A dump occurs automatically upon an abend.

The dump is distributed among the various active services, so which options are actually recognized depends upon the implementation of the other services.

**Syntax:**

```
DUMP TRANSACTION DUMPCODE(name)
    [COMPLETE]
    [FROM(data-area) [LENGTH(data-value) | FLENGTH(data-value)]]
    [TRT]
    [SEGMENTLIST(data-area) LENGTHLIST(data-area)
    NUMSEGMENTS(data-area)]
    [TASK]
    [STORAGE]
    [PROGRAM]
    [TERMINAL]
    [TABLES]
    [DCT]
    [FCT]
    [PCT]
```

[PPT]

[SIT]

[TCT]

### Service:

Dump Control

### Setup:

None

### Options:

#### **DUMPCODE(name)**

Specify a dumpcode signature for the dump.

#### **COMPLETE**

Include all information in the dump.

#### **FROM(data-area)**

Specify a data-area to be dumped along with the remainder of the dump.

#### **LENGTH(data-value)**

The length of the FROM.

#### **FLENGTH(data-value)**

The length of the FROM.

#### **SEGMENTLIST(data-area)**

#### **LENGTHLIST(data-area)**

#### **NUMSEGMENTS(data-area)**

Segment dumping is not currently supported.

This specifies that there are NUMSEGMENTS segments, each listed in SEGMENTLIST with a length specified by LENGTHLIST.

#### **TASK**

Storage associated with task.

#### **STORAGE**

Dump storage.

#### **PROGRAM**

Program storage areas.

#### **TERMINAL**

Storage associated with terminal.

**TABLES - DCT, FCT, PCT, PPT, SIT, TCT.**

**DCT**

Destination Control Table.

**FCT**

File Control Table.

**PCT**

Process Control Table.

**PPT**

Processing Program Table.

**SIT**

System Initialization Table.

**TCT**

Terminal Control Table.

## **ENDBR**

---

End the browsing operation on a file. The browse must previously have been started using STARTBR.

**Syntax:**

```
ENDBR
      {FILE(data-value) | DATASET(data-value)}
      [REQID(data-value)]
      [SYSID(data-value)]
```

**Service:**

File Control

**Setup:**

See file setup under READ.

**Options**

**FILE(data-value)**

Specifies the filename. The filename must be setup to point to an .xml file descriptor.

**DATASET(data-value)**

Specifies the filename, just as does FILE. FILE is preferred.

**REQID(data-value)**

Specifies a previously specified request identifier to control multiple browses.

**SYSID(data-value)**

Specifies the system ID, 1 to 4 characters, of the browse operation.

**Conditions:**

FILENOTFOUND	The file was not found.
ILLOGIC	Logical error.
INVREQ	Invalid request.
IOERR	Input/Output error.
ISCINVREQ	Remote system error.
NOTAUTH	Resource security check failed.
SYSIDERR	System ID error.

## ENQ

---

Acquire a resource for exclusive use.

This command is cooperative, so any other tasks wanting the same resource must also allocate it for exclusive use using the same mechanism.

As this command by definition forces tasks to wait or serialize their actions, it should be avoided wherever possible.

The default implementation uses JNDI. A resource is locked to a single task by binding an object to the JNDI name 'Its\_resource\_<resourcename>', where <resourcename> is the name of the resource to its given length. The resource is unlocked by unbinding the object. (Nesting the enq/deq is left to the individual task when locked between tasks.) The scope of the resource allocation is therefore the scope of JNDI in use, either a single application server or clustered application servers using clustered JNDI.

**Syntax:**

```
ENQ RESOURCE(data-area) [LENGTH(data-value)]
    [UOW | TASK | MAXLIFETIME(cvda)]
    [NOSUSPEND]
    [RETRIES(data-value)]
    [RETRYINTERVAL(data-value)]
```

## Service:

Task Control

## Options:

### **RESOURCE(data-value)**

Specifies the name of the resource as a data-area. This is always the content of the specified resource, not the address of the resource. LENGTH should always be specified, but the implicit length of the resource is used if LENGTH is left unspecified.

### **LENGTH(data-value)**

Specifies the length of the resource name. If left unspecified, an implicit length of the resource data-area is used, not the address of the resource. This must be between 1 and 255.

### **UOW**

Specifies that the resource is to live until the end of the unit of work. This is the default.

### **TASK**

Specifies that the resource is to live until the end of the task.

### **MAXLIFETIME(cvda)**

Specifies the maximum lifetime of the resource, either the CVDA for UOW or TASK. The CVDA LUW is accepted as a synonym for UOW. Specify the lifetime in this way if not constant, otherwise use the UOW or TASK tag directly.

### **NOSUSPEND**

Do not suspend the task waiting for the resource; issue ENQBUSY instead.

### **RETRIES(data-value)**

Specifies a maximum number of retries to obtain the resource if the implementation may use it; it is ignored otherwise.

### **RETRYINTERVAL(data-value)**

Specifies an interval between retries to obtain the resource if the implementation may use it; it is ignored otherwise. The interval is in microseconds, so a RETRYINTERVAL(1000) would delay for one second before retrying.

## Conditions:

ENQBUSY	The resource is busy and cannot be enqueued. This is only issued if not instructed to wait for the resource indefinitely. Unlike most conditions, it is ignored by default, so always code a RESP or HANDLE to catch
---------	--

	this condition.
--	-----------------

INVREQ	
2	MAXLIFETIME was set to an invalid CVDA

LENGERR	
1	LENGTH was not between 1 and 255.

## ENTER TRACENUM

---

Append a record to the trace journal.

### Syntax:

```

ENTER TRACENUM(data-value)
      FROM (data-area) [FROMLENGTH(data-area)]
      [RESOURCE(name)]
      [EXCEPTION]

```

### Service:

Trace Control

### Options:

#### TRACENUM

Trace identifier, a number within 0 through 199.

#### FROM

Contents of the trace entry.

#### FROMLENGTH

Length of the trace entry contents, within 0 through 4000.

#### RESOURCE

Resource field of trace entry.

#### EXCEPTION

Write entry even if user trace flag is off. The text '\*EXCU' is included in the trace entry.

### Conditions:

INVREQ	
1	TRACENUM is not within 0 through 199.

2	No trace destination available.
3	User trace flag is OFF, and EXCEPTION not specified.

LENGERR	
4	FROMLENGTH is not within 0 through 4000.

## EXTRACT CERTIFICATE

---

(THIS COMMAND IS UNSUPPORTED)

Extract information about the client X.509 certificate received during SSL handshake, if present.

### Syntax:

```

EXTRACT CERTIFICATE(ptr-ref)
    [LENGTH(data-area)]
    [SERIALNUM(ptr-ref)]
    [SERIALNUMLEN(data-area)]
    [USERID(ptr-ref)]
    [OWNER | ISSUER]
    [COMMONNAME(ptr-ref)]
    [COMMONNAMLEN(data-area)]
    [COUNTRY(ptr-ref)]
    [STATE(ptr-ref)]
    [STATELEN(data-area)]
    [LOCALITY(ptr-ref)]
    [LOCALITYLEN(data-area)]
    [ORGANIZATION(ptr-ref)]
    [ORGANIZATLEN(data-area)]
    [ORGUNIT(ptr-ref)]
    [ORGUNITLEN(data-area)]

```

### Service:

TCP/IP

### Setup:

None

## Options:

### **CERTIFICATE**

Retrieve the full certificate binary.

### **LENGTH**

Retrieve the length of the full certificate binary.

### **SERIALNUM**

Retrieve the pointer to the serial number.

### **SERIALNUMLEN**

Retrieve the length of the serial number.

### **USERID**

Retrieve the user identification.

### **OWNER**

The extracted values of this command relate to the owner of the certificate. This is the default, if ISSUER is not specified.

### **ISSUER**

The extracted values of this command relate to the issuer of the certificate.

### **COMMONNAME**

Retrieve the common name from the client certificate.

### **COMMONNAMLEN**

Retrieve the length of the common name retrieved.

### **COUNTRY**

Retrieve the pointer to the country.

### **STATE**

Retrieve the pointer to the state or province.

### **STATELEN**

Retrieve the length of the state or province returned.

### **LOCALITY**

Retrieve the locality.

### **LOCALITYLEN**

Retrieve the length of the locality.

### **ORGANIZATION**

Retrieve the organization.

### **ORGANIZATLEN**

Retrieve the length of the organization.

**ORGUNIT**

Retrieve the organizational unit.

**ORGUNITLEN**

Retrieve the length of the organizational unit.

**Conditions:**

INVREQ	
	Occurs for the following reasons: The common is not in a web session, Not in an HTTP request, There is no certificate, Or there is an error in retrieving the certificate.

LENGERR	The extracted contents exceed the corresponding length.
---------	---

## EXTRACT TCPIP

---

Extract information about the current TCP/IP connection.

The current implementation is available only when executing as a web session.

**Syntax:**

EXTRACT TCPIP

```
[CLIENTNAME(data-area) CNAMELENGTH(data-area)]
[SERVERNAME(data-area) SNAMELENGTH(data-area)]
[CLIENTADDR(data-area) CADDRLENGTH(data-area)]
[CLIENTADDRNU(data-area)]
[SERVERADDR(data-area) SADDRLENGTH(data-area)]
[SERVERADDRNU(data-area)]
[SSLTYPE(cvda)]
[TCPIPSERVICE(data-area)]
[PORTNUMBER(data-area)]
[PORTNUMNU(data-area)]
```

**Service:**

TCP/IP

## Setup:

None

## Options:

### **CLIENTNAME**

Extract the client host name, address in text form if unknown.

### **CNAMELENGTH**

Pass the length of the CLIENTNAME buffer, receive the length of contents.

### **SERVERNAME**

Extract the server host name, address in text form if unknown.

### **SNAMELENGTH**

Pass the length of the SERVERNAME buffer, receive the length of contents.

### **CLIENTADDR**

Receive the client address in text form.

### **CADDRLENGTH**

Pass the length of the CLIENTADDR buffer, receive the length of contents.

### **CLIENTADDRNU**

Receive the client address in four-byte form, suitable for PIC X(2) COMP-X.

### **SERVERADDR**

Receive the server address in text form.

### **SADDRLENGTH**

Pass the length of the SERVERADDR buffer, receive the length of contents.

### **SERVERADDRNU**

Receive the server address in four-byte form, suitable for PIC X(2) COMP-X.

### **SSLTYPE**

The CVDA value of SSL, NOSSL, or CLIENTAUTH, checkable against DFHVALUE(SSL), DFHVALUE(NOSSL) or DFHVALUE(CLIENTAUTH) in program code.

NOSSL is returned when connected without SSL, as in the http protocol.

SSL is returned when connected using SSL, as in the https protocol.

CLIENTAUTH is returned when connected using SSL and the client has client authorization, a client certificate.

### **TCPIPSERVICE**

Extract the TCP/IP service name, such as 'HTTP/1.1'.

### **PORTNUMBER**

Extract the port number of the TCP/IP service in text form.

**PORTNUMNU**

Extract the port number of the TCP/IP service in numeric form.

**Conditions:**

INVREQ	The session is not a web session.
--------	-----------------------------------

LENGERR	The length extracted is longer than the length allowed by the corresponding LENGTH parameter.
---------	---

## **EXTRACT WEB**

---

See WEB EXTRACT.

## **FORMATTIME**

---

Format a given absolute date/time timestamp into human readable formats.

**Syntax:**

```
FORMATTIME [ABSTIME(data-area)]
           [DATE(data-area)]
           [FULLDATE(data-area)]
           [DATEFORM(data-area)]
           [DATESEP | DATESEP(data-value)]
           [DAYCOUNT(data-area)]
           [DAYOFMONTH(data-area)]
           [DAYOFWEEK(data-area)]
           [DDMMYY(data-area)]
           [DDMMYYYY(data-area)]
           [MMDDYY(data-area)]
           [MMDDYYYY(data-area)]
           [MONTHOFYEAR(data-area)]
           [TIME(data-area) [TIMESEP | TIMESEP(data-value)]]
           [YEAR(data-area)]
           [YYDDD(data-area)]
           [YYDDMM(data-area)]
           [YYMMDD(data-area)]
```

[YYYYDDD(data-area)]  
[YYYYDDMM(data-area)]  
[YYYYMMDD(data-area)]

## Service:

Date Time

## Setup:

init.dateform

This should be set to one of the following values:

YYMMDD (default)  
DDMMYY  
MMDDYY

## Options:

In all options where date formatting is returned, M is month digit, D is date digit, Y is year digit.

### **ABSTIME(data-area)**

Data-area is a numeric, specifying the number of milliseconds since January 1, 1900 at 00:00, as returned by ASKTIME. This is the value used as the basis for other formatting options. If not specified, the current date/time timestamp is used.

### **DATE(data-area)**

Retrieve date in form specified by init.datform, where the date separator is specified by DATESEP (blank otherwise).

### **FULLDATE(data-area)**

### **DATEFORM(data-area)**

Retrieve the content of init.datform.

### **DATESEP(data-value)**

If data-value is specified, it is the date separator; if data-value is not specified, '/' is used.

Data-value must be one character in length.

### **DAYCOUNT(data-area)**

Retrieve number of days since January 1, 1900.

### **DAYOFMONTH(data-area)**

Retrieve number of day in the month.

### **DAYOFWEEK(data-area)**

Retrieve number of day in the week.

**DDMMYY(data-area)**

Retrieve the date in DDMMYY format.

**DDMMYYYY(data-area)**

Retrieve the date in DDMMYYYY format.

**MMDDYY(data-area)**

Retrieve the date in MMDDYY format.

**MMDDYYYY(data-area)**

Retrieve the date in MMDDYYYY format.

**MONTHOFYEAR(data-area)**

Retrieve the month of the year.

**TIME(data-area)**

Retrieve the time, where the separator in hh:mm:ss is specified by TIMESEP (blank otherwise).

**TIMESEP(data-value)**

If data-value is specified, it is the time separator; if data-value is not specified, ':' is used.

Data-value must be one character in length.

**YEAR(data-area)**

Retrieve the year.

**YYDDD(data-area)**

Retrieve the date in YYDDD format.

**YYDDMM(data-area)**

Retrieve the date in YYDDMM format.

**YYMMDD(data-area)**

Retrieve the date in YYMMDD format.

**YYYYDDD(data-area)**

Retrieve the date in YYYYDDD format.

**YYYYDDMM(data-area)**

Retrieve the date YYYYDDMM format.

**YYYYMMDD(data-area)**

Retrieve the date in YYYYMMDD format.

**Conditions:**

None

## FREEMAIN

---

Free memory allocated by GETMAIN.

In this environment, memory is garbage collected automatically, so FREEMAIN does nothing. If additional pointers point to the memory, it will remain allocated. Additionally, if no pointers are pointing to the memory it is removed automatically.

### Syntax:

```
FREEMAIN  
    {DATA(data-area) | DATAPOINTER(ptr-value)}
```

### Setup:

None

### Service:

Storage Control

### Options:

#### **DATA(data-area)**

Specifies the main storage to be released.

#### **DATAPOINTER(ptr-value)**

Specifies the main storage to be released.

### Conditions:

None

## GET COUNTER

---

Get the next value from the counter service.

### Syntax:

```
GET { COUNTER(name) | DCOUNTER(name) }  
    [POOL(name)]  
    VALUE(data-area)
```

[INCREMENT(data-value) [REDUCE]]  
[WRAP]  
[COMPAREMIN(data-value)]  
[COMPAREMAX(data-value)]

## Service:

Counter

## Setup:

**pool.name=redirected\_pool\_name**

If not specified, the pool name is the given name.

**counter.jndi=counter\_jndi\_name**

Specify the JNDI name of the Counter service. The default is 'CounterService'.

## Options:

**COUNTER(name)**

Specifies the name of the 32-bit counter.

**DCOUNTER(name)**

Specifies the name of the 64-bit counter.

**POOL(name)**

Specifies the name of the pool of counters. Each Counter service maintains its own pools of counters.

**VALUE(data-area)**

Retrieves the value of the counter.

**INCREMENT(data-value)**

Specifies an alternate increment value, rather than one (1).

**REDUCE**

Specifies that the increment is to be reduced if the increment would force the counter to be at its limit. The result would then be the maximum value plus one (1).

**WRAP**

Automatically performs a wrap rather than notify that the counter is its limit.

**COMPAREMIN(data-value)**

Specifies a minimum value against which to compare the value. If the value is greater than or equal, then no action is taken; if less than the minimum, a condition is raised.

If COMPAREMIN>COMPAREMAX, then either condition must be true rather than both.

Success of the command is conditional upon the comparison.

**COMPAREMAX(data-value)**

Specifies a maximum value against which to compare the value. If the value is less than or equal, then no action is taken; if less than the minimum, a condition is raised.

If COMPAREMIN>COMPAREMAX, then either condition must be true rather than both.

Success of the command is conditional upon the comparison.

**Conditions:**

INVREQ	RESP2
201	Named counter invalid.
301	Server error.
303	Unexpected error, such as connectivity loss.
304	Invalid pool.
305	Cannot connect to server.
306	Serverabend.
308	This condition cannot occur (options table not loadable).
309	This condition cannot occur (options table error).
310	This condition cannot occur (options user exit).
403	The pool name contains invalid characters.
404	The counter name contains invalid characters.
406	The increment is invalid; it cannot be larger than the total range of the counter.

LENGERR	LENGERR occurs only for COUNTER commands, not DCOUNTER commands.
001	The value has become negative.
002	The value is too large by one bit.
003	The value is too large by more than one bit.

SUPPRESSED	RESP2
101	The maximum value has already been assigned. It must be reset or retrieved using WRAP.
103	The value is not within COMPAREMIN and COMPAREMAX, or beyond the limits if only one of COMPAREMIN and COMPAREMAX is used

# GETMAIN

---

Allocate main storage, returning a pointer to it. FREEMAIN is then used to free the memory afterwards.

## Syntax:

```
GETMAIN SET(ptr-ref)
        {{FLENGTH(data-value) [BELOW]} | (LENGTH(data-value))}
        [INITIMG(data-value)]
        [SHARED]
        [NOSUSPEND]
        [USERDATAKEY | CICS DATAKEY]
```

## Setup:

None

## Service:

Storage Control

## Options:

### **SET(ptr-ref)**

SET retrieves the block of allocated storage.

### **FLENGTH(data-value)**

FLENGTH determines the length of allocated storage. FLENGTH is preferred over LENGTH.

### **BELOW**

The BELOW option is to specify the location of allocated memory. This option has no meaning with the environment and is ignored.

### **LENGTH(data-value)**

LENGTH determines the length of allocated storage. FLENGTH is preferred over LENGTH.

### **INITIMG(data-value)**

Specify a one-character initialization for allocated memory. By default, memory is initialized to low-values / nulls.

### **SHARED**

The SHARED option allocates the memory for shared storage. This is not possible within the environment, so it always raises the condition UNSUPPORTED\_OPTION.

### **NOSUSPEND**

Do not suspend the task if memory is not available. This continuously waits and retries the allocation.

### **USERDATAKEY**

This specifies the location of allocated memory and has no meaning within the environment. It is ignored.

### **CICSDATAKEY**

This specifies the location of allocated memory and has no meaning within the environment. It is ignored.

### **Conditions:**

LENGERR	RESP2
1	The FLENGTH < 1, or LENGTH = 0.
NOSTG	RESP2
2	The storage requested is not currently available.
UNSUPPORTED_OPTIO N	This is always raised by SHARED.

## **JOURNAL**

---

This command is obsolete, replaced by WRITE JOURNALNAME. See WRITE JOURNALNUM for information on the obsolete syntax.

## **HANDLE ABEND**

---

Handle an abnormal termination by executing procedural code or another program.

### **Syntax:**

```
HANDLE ABEND  
    {CANCEL | RESET | PROGRAM(name) | LABEL(label)}
```

## Service:

Program Control

## Options:

### **CANCEL**

Specifies that any previous handle for abend should be cancelled.

### **RESET**

Specifies that a previously cancelled abend should be reinstated.

### **LABEL(label)**

Specifies a paragraph or section label to execute upon abend.

### **PROGRAM(name)**

Specifies a program name to link and execute upon abend.

## Conditions:

PGMIDERR	The program could not be linked. This may be delayed until the actual abend occurs.
----------	---

# HANDLE AID

---

Handle Attention Identifier instructs the program to automatically transfer control to the given label when the Attention Identifier key is recognized as pressed in the RECEIVE MAP command.

## Syntax:

```
HANDLE  
    ANYKEY(label)  
    CLEAR(label)  
    CLRPARTN(label)  
    ENTER(label)  
    LIGHTPEN(label)  
    OPERID(label)  
    PA1...PA3(label)  
    PF1...PA24(label)  
    TRIGGER(label)
```

**Service:**

Execute Interface Program

**Setup:**

None

**Options:****key(label)**

Label specifies a paragraph or section name. If label is omitted, then the default handler is restored.

**Conditions:**

None

## HANDLE CONDITION

---

Handle conditions explicitly using custom program logic rather than the default condition handler (generally the termination dump). These handlers may be suspended and restored using PUSH HANDLE and POP HANDLE.

**Syntax:**

```
HANDLE CONDITION {condition(label)}...
```

**Service:**

Execute Interface Program

**Setup:**

None

**Options:****condition(label)**

Condition specifies a condition name, such as INVREQ, QIDERR, ERROR, etc.

Label specifies a paragraph or section name. If label is omitted, then the default handler is restored.

**Conditions:**

None

## IGNORE CONDITION

---

Ignore conditions explicitly rather than executing the default condition handler (generally the termination dump) or other custom handler. These handlers may be suspended and restored using PUSH HANDLE and POP HANDLE.

**Syntax:**

```
IGNORE CONDITION {condition}...
```

**Service:**

Execute Interface Program

**Setup:**

None

**Options:****condition**

Condition specifies a condition name, such as INVREQ, QIDERR, ERROR, etc.

**Conditions:**

None

## LINK

---

Link to another program, and then return continuing the existing program.

**Syntax:**

```
LINK PROGRAM(name)  
      [COMMAREA(data-area) [LENGTH(data-area)] [DATALENGTH(data-  
value)]  
      [INPUTMSG(data-area) [INPUTMSGLEN(data-value)]]
```

[SYSID(name)]  
[SYNCONRETURN]  
[TRANSID(name)]

### Service:

Program Control

### Setup:

#### **pathname=program Uri**

program\_name must be a valid program reference, either a class name or URI.

It may be a class-name/program-name, or remote:jndi-name.

#### **pct.name.sysid=jndi\_name**

This establishes the sysid for a program, when one is not specified in the link.

#### **pct.name.transid=transid\_name**

This establishes the default transid for a program, when one is not specified in the link.

#### **sysid.name=jndi\_name**

This establishes a JNDI name for the sysid, possibly referencing a remote EJB.

#### **xlt.name=program\_name**

program\_name must be a valid name in the program table.

### Options:

#### **PROGRAM(name)**

The program name to be loaded.

#### **COMMAREA(data-area)**

The COMMAREA to pass to the linked program.

#### **LENGTH(data-area)**

The length of the COMMAREA.

#### **DATALENGTH(data-value)**

The data length of the COMMAREA to pass to the other program. This must be less than or equal to the LENGTH.

#### **INPUTMSG(data-area)**

Specifies data to be received by the other program's RECEIVE.

**INPUTMSGLEN(data-value)**

Specifies the length of the INPUTMSG.

**SYSID(name)**

Specifies the system on which the program is to be linked. This means that the program name's table entry begins with the system id followed by a period, then the normal program name.

**SYNCONRETURN**

Only meaningful for Distributed Program Link, specifies that the linked program is to take a syncpoint upon completion.

**TRANSID(name)**

Specifies the transaction name to link.

**Conditions:**

INVREQ	
16	TRANSID is all blanks.

LENGERR	
11	COMMAREA length not between 0 and 32767
12	DATALENGTH < 0
13	DATALENGTH > LENGTH
27	INPUTMSGLEN not between 0 and 32767

PGMIDERR	
1	The program name has no table entry.
3	The program could not be loaded.

## LOAD

---

Load a program or resource into memory.

As the target environment provides a strong separation between program data and program code, whether ENTRY or SET is used to obtain the return reference determines the type of load. If ENTRY, program code is returned; if SET, then program data is returned.

**Syntax:**

```
LOAD {PROGRAM(name) | RESOURCE(name)}
      [SET(pointer-ref)]
      [ENTRY(procedure-pointer-ref)]
```

[LENGTH(data-area) | FLENGTH(data-area)]  
[HOLD]

## Service:

Program Control

## Setup:

### **rct.name=resource\_name**

resource\_name must be the name of a data file in the resources directory, or a classname representing a java.lang.Runnable class, such as an Elastic COBOL program.

### **pct.name=program\_uri**

program\_uri must be a valid program reference, either a classname or URI.

## Options:

### **PROGRAM(name)**

The program name to be loaded.

### **RESOURCE(name)**

The resource name to be loaded. This is a synonym for PROGRAM, preferred for data. It is only referenced if PROGRAM is not present.

### **ENTRY(procedure-pointer-ref)**

This returns a reference to a Java executable class. This will use the resource table name, if available, otherwise the program table name.

### **LENGTH(data-area)**

Only used for data, returns the length of the data area set.

### **FLENGTH(data-area)**

Only used for data, returns the length of the data area set.

### **SET(pointer-ref)**

Returns the memory containing the data represented by the program or resource name.

### **HOLD**

Requests that the data area not be released when the task is ended. As this system uses a garbage collection scheme for memory allocation, this option has no effect. The memory is released automatically whenever no references are made to it.

**Conditions:**

PGMIDERR	
1	The program name has no table entry.
3	The program could not be loaded.

## MONITOR

---

(THIS COMMAND IS UNSUPPORTED)

MONITOR provides performance information.

**Syntax:**

MONITOR

POINT(data-value) [DATA1(data-area) DATA2(data-area)  
ENTRYNAME(data-area)]

**Service:**

Trace Control

## POP HANDLE

---

Restore condition handlers from the stack, previously pushed there by PUSH HANDLE. This is useful if desiring the suspension of handling for a routine and it should be paired with PUSH HANDLE.

**Syntax:**

POP HANDLE

**Service:**

Execute Interface Program

**Setup:**

None

**Options:**

None

**Conditions:**

INVREQ	
	Indicates that there was no matching PUSH HANDLE.

## POST

---

Post a message to an area of memory after a specified delay. The task continues processing while the post waits to perform its action.

INTERVAL or AFTER specify a period of time, whereas TIME or AT specify the final time itself.

Note that most foreground sessions contain timeouts and will be terminated if delayed too far.

**Syntax:**

```

POST
    SET(pointer-ref)
    [REQID(data-value)]
        [INTERVAL(hhmmss)]
    [AFTER
    [HOURS(data-value)]
    [MINUTES(data-value)]
    [SECONDS(data-value)]
    ]
    [TIME(hhmmss)]
    [AT
        [HOURS(data-value)]
        [MINUTES(data-value)]
        [SECONDS(data-value)]
    ]

```

**Service:**

Interval Control

**Setup:**

None

## Options:

### **SET(pointer-ref)**

Specifies a pointer (USAGE POINTER) data item which is set to a four-byte memory control area. When the delay is finished and the post is done, the first and third byte are altered to X"40" and X"80". This exact same pointer item may be passed to the WAIT EVENT command.

### **REQID(data-value)**

Specifies a request ID through which this command may be cancelled using the CANCEL command. The request ID is valid only within the same session.

### **INTERVAL(hhmmss)**

Specifies an interval of time duration data-value. The format is a single number with two-digits each hour, minute, second.

### **TIME(hhmmss)**

Specifies a final time data-value. The format is a single number with two-digits each hour, minute, second.

### **AFTER**

AFTER specifies that HOURS, MINUTES and SECONDS refers to an interval duration.

### **AT**

AT specifies that HOURS, MINUTES and SECONDS refers to a final time.

### **HOURS(data-value)**

Specifies the number of hours, and it must be within 0..99.

### **MINUTES(data-value)**

Specifies the number of minutes, and it must be within 0..59 if other HOURS or SECONDS is specified.

### **SECONDS(data-value)**

Specifies the number of seconds, and it must be within 0..59 if other HOURS or MINUTES is specified.

## Conditions:

EXPIRED	The interval has already expired.
INVREQ	RESP2
	4 Hours are invalid. 5 Minutes are invalid. 6 Seconds are invalid.

## PURGE MESSAGE

---

(THIS COMMAND IS UNSUPPORTED)  
Purge the current BMS logical message.

**Syntax:**

PURGE MESSAGE

**Service:**

Basic Mapping Support

**Setup:**

None

**Options:**

None

**Conditions:**

INVREQ	Invalid request.
TSIOERR	Temporary storage input/output error.

## PUSH HANDLE

---

Push the current condition handlers onto the stack. This is useful if desiring the suspension of handling for a routine and it should be paired with POP HANDLE.

**Syntax:**

PUSH HANDLE

**Service:**

Execute Interface Program

**Setup:**

None

### Options:

None

### Conditions:

None

## QUERY COUNTER

---

Query attributes of the named counter.

### Syntax:

```
UPDATE { COUNTER(name) | DCOUNTER(name) } [POOL(name)]  
      [VALUE(data-area)]  
      [MINIMUM(data-area)]  
      [MAXIMUM(data-area)]
```

### Service:

Counter

### Setup:

#### **pool.name=redirected\_pool\_name**

If not specified, the pool name is the given name.

#### **counter.jndi=counter\_jndi\_name**

Specify the JNDI name of the Counter service. The default is 'CounterService'.

### Options:

#### **COUNTER(name)**

Specifies the name of the 32-bit counter.

#### **DCOUNTER(name)**

Specifies the name of the 64-bit counter.

#### **POOL(name)**

Specifies the name of the pool of counters. Each Counter service maintains its own pools of counters.

**VALUE(data-area)**

Retrieves the current value of the counter. The counter's value is only retrieved, not updated, by this action.

**MINIMUM(data-area)**

Retrieves the minimum value of the counter, as given upon its definition.

**MAXIMUM(data-area)**

Retrieves the maximum value of the counter, as given upon its definition.

**Conditions:**

INVREQ	RESP2
201	Named counter invalid.
301	Server error.
303	Unexpected error, such as connectivity loss.
304	Invalid pool.
305	Cannot connect to server.
306	Server abend.
308	This condition cannot occur (options table not loadable).
309	This condition cannot occur (options table error).
310	This condition cannot occur (options user exit).
403	The pool name contains invalid characters.
404	The counter name contains invalid characters

LENGERR	LENGERR occurs only for COUNTER commands, not DCOUNTER commands.
001	The value has become negative.
002	The value is too large by one bit.
003	The value is too large by more than one bit.

## QUERY SECURITY

---

This command is dependent upon the capabilities of the External Security Manager.

**Syntax:**

```

QUERY SECURITY
    {RESTYPE(data-value) | RESCLASS(data-value) RESIDLENGTH(data-
value)}
    RESID(data-value)
    [LOGMESSAGE(cvda)]
    [READ(cvda)]

```

[UPDATE(cvda)]  
[CONTROL(cvda)]  
[ALTER(cvda)]

### Service:

Security

### Setup:

None

### Options:

#### **ALTER(cvda)**

Retrieves whether user has alter authority for resource, returning ALTERABLE or NOTALTERABLE.

#### **CONTROL(cvda)**

Retrieves whether user has control authority for resource, returning CTRLABLE or NOTCTRLABLE.

#### **LOGMESSAGE(cvda)**

Specifies whether security violation messages are logged, passing LOG or NOLOG.

#### **READ(cvda)**

Retrieves whether user has read authority for resource, returning READABLE or NOTREADABLE.

#### **RESCLASS(data-value)**

Specifies 8-character resource class. SPCOMMAND refers to an ETP resource.

#### **RESID(data-value)**

Specifies the resource ID to check.

#### **RESIDLENGTH(data-value)**

Specifies the length of the resource ID to check.

#### **RESTYPE(data-value)**

Specifies the type of resource, 1 to 12 characters, from the list:

<b>DB2ENTRY</b>	<b>SPCOMMAND</b>	<b>FILE</b>
<b>TDQUEUE</b>	<b>JOURNALNAME</b>	<b>TRANSACTION</b>
<b>JOURNALNUM</b>	<b>TRANSATTACH</b>	<b>PROGRAM</b>
<b>TSQUEUE</b>	<b>PSB</b>	

## UPDATE(cvda)

Retrieves whether user has update authority for resource, returning UPDATABLE or NOTUPDATABLE.

### Conditions:

INVREQ	
LENGERR	
NOTFND	
QIDERR	

## READ

---

Read a single record from the file.

### Syntax:

```
READ {FILE(data-value) | DATASET(data-value)}  
    [UPDATE [TOKEN(data-area)]  
    | UNCOMMITTED  
    | CONSISTENT  
    | REPEATABLE  
    ]  
    {INTO(data-area) | SET(pointer-ref)}  
    RIDFLD(data-area) [KEYLENGTH(data-value) [GENERIC]]  
    [SYSID(data-value) LENGTH(data-area) | LENGTH(data-area)]  
    [RBA | RRN | DEBKEY | DEBREC]  
    [EQUAL | GTEQ]  
    [NOSUSPEND]
```

### Service:

File Control

### Setup:

```
file.<filename>=jdbc:< xml_file_descriptor.xml>;<sql_connection_name>
```

The `xml_file_descriptor` is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a `$XFD FILE="filename.xml"` command before a record in the data division will generate a starting template in the listing folder. The `sql_connection_name` is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

## Options:

### **CONSISTENT**

Specifies data integrity, the record is read in a consistent manner with other data clients.

### **DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

### **DEBKEY**

Deblocking by key in BDAM, this option is not used.

### **DEBREC**

Deblocking by record in BDAM, this option is not used.

### **EQUAL**

Specifies that the record's key must be equal to the given record ID field.

### **FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.

### **GENERIC**

Specifies that the record ID field is generic, matching any with the given contents equal up the length of keylength.

### **GTEQ**

Specifies that the record's key must be greater than or equal to the given record ID field.

### **INTO(data-area)**

Specifies the data-area into which the record is read.

### **KEYLENGTH(data-value)**

Specifies the length of the record ID key field.

### **LENGTH(data-area)**

Specifies the length of the into field. Upon return, this may indicate the length of the actual retrieved record. As a mapping is occurring between records and SQL, the return length will generally be the maximum record length.

### **NOSUSPEND**

The read does not suspend on locks.

**RBA**

Specifies that the key is by relative byte address.

**REPEATABLE**

Specifies data integrity, the record read is repeatable.

**RIDFLD(data-area)**

Specifies the record ID field, the key used to search for the record. This data item is mapped to the column marked with a ridfld attribute in the XML file descriptor.

**RRN**

Specifies that the key is by relative record number.

**SET(pointer-ref)**

Retrieves the data record into a data area pointed to by the pointer ref.

**SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

**TOKEN(data-area)**

Specifies a token to be used later in a rewrite command.

**UNCOMMITTED**

Specifies data integrity, the record read is uncommitted.

**UPDATE**

Specifies that the record is to be read for update, necessary for the rewrite command.

**Conditions:**

DUPKEY	The record has a duplicate key that has not yet been read.
FILENOTFOUND	The file itself could not be found.
ILLOGIC	A logical error occurred.
INVREQ	Invalid request.
IOERR	An input/output error occurred, such as an invalid SQL query.
LENGERR	The length of the record read exceeds the given INTO buffer's LENGTH.
NOTAUTH	Not authorized.
NOTFND	The record was not found, such as an invalid record ID field.
NOTOPEN	The file is not open.

RECORDBUSY	The record was locked.
------------	------------------------

# READNEXT

---

Read the next record from the file during a browse operation. This must be bracketed by STARTBR and ENDBR commands.

**Syntax:**

```

READNEXT {FILE(data-value) | DATASET(data-value)}
        [UPDATE [TOKEN(data-area)]
        | UNCOMMITTED
        | CONSISTENT
        | REPEATABLE
        ]
        {INTO(data-area) | SET(pointer-ref)}
        RIDFLD(data-area) [KEYLENGTH(data-value) [REQID(data-value)]]
        [SYSID(data-value) LENGTH(data-area) | LENGTH(data-area)]
        [RBA | RRN]
        [NOSUSPEND]

```

**Service:**

File Control

**Setup:**

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**  
 The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**  
 The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

**Options:**

**CONSISTENT**

Specifies data integrity, the record is read in a consistent manner with other data clients.

**DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

**FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.

**INTO(data-area)**

Specifies the data-area into which the record is read.

**KEYLENGTH(data-value)**

Specifies the length of the record ID key field.

**LENGTH(data-area)**

Specifies the length of the into field. Upon return, this may indicate the length of the actual retrieved record. As a mapping is occurring between records and SQL, the return length will generally be the maximum record length.

**NOSUSPEND**

The read does not suspend on locks.

**RBA**

Specifies that the key is by relative byte address.

**REPEATABLE**

Specifies data integrity, the record read is repeatable.

**REQID(data-value)**

Specifies a browser's request ID.

**RIDFLD(data-area)**

Specifies the record ID field, the key used to search for the record. This data item is mapped to the column marked with an ridfld attribute in the XML file descriptor.

**RRN**

Specifies that the key is by relative record number.

**SET(pointer-ref)**

Retrieves the data record into a data area pointed to by the pointer ref.

**SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

**TOKEN(data-area)**

Specifies a token to be used later in a rewrite command.

### **UNCOMMITTED**

Specifies data integrity, the record read is uncommitted.

### **UPDATE**

Specifies that the record is to be read for update, necessary for the rewrite command.

### **Conditions:**

DUPKEY	The record has a duplicate key that has not yet been read.
ENDFILE	The end of the browse request's result set has been reached.
FILENOTFOUN D	The file itself could not be found.
ILLOGIC	A logical error occurred.
INVREQ	Invalid request.
IOERR	An input/output error occurred, such as an invalid SQL query.
LENGERR	The length of the record read exceeds the given INTO buffer's LENGTH.
NOTAUTH	Not authorized.
NOTFND	The record was not found, such as an invalid record ID field.
NOTOPEN	The file is not open.
RECORDBUSY	The record was locked.

## **READPREV**

---

Read the previous record from the file during a browse operation. This must be bracketed by STARTBR and ENDBR commands.

Note, not all SQL JDBC drivers support the operation required to browse previous records. An IOERR with a message about an operation not being supported by JDBC 2.0 will be given if the JDBC driver does not have sufficient support for this command.

### **Syntax:**

```
READPREV {FILE(data-value) | DATASET(data-value)}  
        [UPDATE [TOKEN(data-area)]  
        | UNCOMMITTED  
        | CONSISTENT  
        | REPEATABLE
```

```
]
{INTO(data-area) | SET(pointer-ref)}
RIDFLD(data-area) [KEYLENGTH(data-value) [REQID(data-value)]]]
[SYSID(data-value) LENGTH(data-area) | LENGTH(data-area)]
[RBA | RRN]
[NOSUSPEND]
```

## Service:

File Control

## Setup:

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**

The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

## Options:

### **CONSISTENT**

Specifies data integrity, the record is read in a consistent manner with other data clients.

### **DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

### **FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.

### **INTO(data-area)**

Specifies the data-area into which the record is read.

### **KEYLENGTH(data-value)**

Specifies the length of the record ID key field.

### **LENGTH(data-area)**

Specifies the length of the into field. Upon return, this may indicate the length of the actual retrieved record. As a mapping is occurring between records and SQL, the return length will generally be the maximum record length.

**NOSUSPEND**

The read does not suspend on locks.

**RBA**

Specifies that the key is by relative byte address.

**REPEATABLE**

Specifies data integrity, the record read is repeatable.

**REQID(data-value)**

Specifies a browser's request ID.

**RIDFLD(data-area)**

Specifies the record ID field, the key used to search for the record. This data item is mapped to the column marked with an ridfld attribute in the XML file descriptor.

**RRN**

Specifies that the key is by relative record number.

**SET(pointer-ref)**

Retrieves the data record into a data area pointed to by the pointer ref.

**SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

**TOKEN(data-area)**

Specifies a token to be used later in a rewrite command.

**UNCOMMITTED**

Specifies data integrity, the record read is uncommitted.

**UPDATE**

Specifies that the record is to be read for update, necessary for the rewrite command.

**Conditions:**

DUPKEY	The record has a duplicate key that has not yet been read.
ENDFILE	The end of the browse request's result set has been reached.
FILENOTFOUN D	The file itself could not be found.

ILLOGIC	A logical error occurred.
INVREQ	Invalid request.
IOERR	An input/output error occurred, such as an invalid SQL query.
LENGERR	The length of the record read exceeds the given INTO buffer's LENGTH.
NOTAUTH	Not authorized.
NOTFND	The record was not found, such as an invalid record ID field.
NOTOPEN	The file is not open.
RECORDBUSY	The record was locked.

## READQ TD

---

Read data from a transient data queue.

### Syntax:

```

READQ TD QUEUE(name)
           [INTO(data-area) | SET(pointer-ref)]
           [LENGTH(data-area)]
           [SYSID(name)]
           [NOSUSPEND]

```

### Service:

Transient Data Control

### Setup:

```

queue[.sysid_value].name=uri
[sysid.name=sysid_value]

```

### Options

#### **QUEUE(name)**

Specifies the name of the transient data queue. The queue must be defined.

#### **INTO(data-area)**

Specifies the data area into which the queue data is copied.

#### **SET(pointer-ref)**

Returns a pointer to the read data record.

**LENGTH(data-value)**

Specifies the length of the INTO buffer, if provided. Returns the length of the returned data.

**SYSID(name)**

Specifies the system ID. This is used to determine which queue definition is used.

**Conditions**

SYSIDERR	An unknown sysid is being used.
QIDERR	
1	There is no such queue definition.
2	The queue could not be obtained.
QBUSY	The queue was busy and could not be read, only possible if NOSUSPEND is specified.
QZERO	The end of the queue has been reached.
INVREQ	The queue consumer could not be created or started, for reasons such as being an output only queue.
IOERR	The receive from queue responded with an error.

## READQ TS

---

Read data from a temporary storage data queue.

**Syntax:**

```

READQ TS {QUEUE(name) | QNAME(name)}
        [INTO(data-area) | SET(pointer-ref)]
        [LENGTH(data-area)]
        [SYSID(name)]
        [NOSUSPEND]
        [NUMITEMS(data-area)]
        [NEXT | ITEM(data-value)]

```

**Service:**

Temporary Storage Control

## Setup:

```
queue[.sysid_value].name[.main | .auxiliary]=uri  
[sysid.name=sysid_value]
```

## Options

### **QUEUE(name)**

Specifies the name of the transient data queue. The queue must be defined.

### **QNAME(name)**

See QUEUE.

### **INTO(data-area)**

Specifies the data area into which the queue data is copied.

### **SET(pointer-ref)**

Returns a pointer to the read data record.

### **LENGTH(data-value)**

Specifies the length of the INTO buffer, if provided. Returns the length of the returned data.

### **SYSID(name)**

Specifies the system ID. This is used to determine which queue definition is used.

### **NUMITEMS(data-area)**

Returns the number of items in the queue.

### **ITEM(data-area)**

Specifies the queue item number to read.

### **NEXT**

Specifies that the next item in the queue is to be read. This position counter is shared at the scope of the queue, which may not be the individual task.

## Conditions

SYSIDERR	An unknown sysid is being used.
QIDERR	
1	There is no such queue definition.
2	The queue could not be obtained.
QBUSY	The queue was busy and could not be read, only possible if NOSUSPEND is specified.

ITEMERR	The requested item is not present in the queue.
---------	---

LENGERR	The length of the read data is longer than the allocated LENGTH. This is only possible when INTO is used rather than SET.
---------	---

## RECEIVE MAP

---

Receive the presentation map's input.

### Syntax:

```
RECEIVE MAP(name)
    [MAPSET(name)]
    [INTO(data-area) | SET(pointer-ref)]
    [
    TERMINAL [ASIS] [INPARTN(data-value)]
    | FROM(data-area) [LENGTH(data-value)]
    ]
```

### Service:

Basic Mapping Support

### Setup:

None

### Options:

#### ASIS

**FROM(data-area)**

**INPARTN(data-value)**

**INTO(data-area)**

**LENGTH(data-value)**

**MAP(name)**

The name is a literal data-value.

**MAPSET(name)**

The name is a literal data-value.

**SET(pointer-ref)**

**TERMINAL**

**Conditions:**

EOC	
EODS	
INVMPSZ	
INVPARTN	
INVREQ	
MAPFAIL	
PARTNFAIL	
RDATT	
UNEXPIN	

## RECEIVE PARTN

---

**Syntax:**

```
RECEIVE PARTN(data-area)
    {INTO(data-area) | SET(pointer-ref)}
    [LENGTH(data-value)]
    [ASIS]
```

**Service:**

Basic Mapping Support

**Setup:**

None

**Options:**

**ASIS**

**INTO(data-area)**

**LENGTH(data-value)**

**PARTN(data-area)**

**SET(pointer-ref)**

**Conditions:**

EOC	
EODS	
INVPARTN	
INVREQ	
LENGERR	

## RELEASE

---

Release a loaded program resource.

This command does nothing in this system, as this system employs garbage collection for automatic release of resources when no longer referenced. The resource is available when any pointers or procedure-pointers referencing it have been altered to point to another item, including null.

It is good practice to null such pointers in all cases as in other systems they would become invalid after release.

**Syntax:**

```
RELEASE PROGRAM(name)
```

**Service:**

Program Control

**Setup:**

None

**Options:****PROGRAM(name)**

The program name is the one to be released.

**Conditions:**

None

## RESETBR

---

Reset a browse operation. This is similar to an ENDBR followed by a STARTBR.

## Syntax:

```
RESETBR
    {FILE(data-value) | DATASET(data-value)}
    [KEYLENGTH(data-value) [GENERIC]]
    [REQID(data-value)]
    [SYSID(data-value)]
    [GTEQ | EQUAL]
    [RBA | RRN]
```

## Service:

File Control

## Setup:

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**

The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

## Options:

### **DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

### **EQUAL**

Specifies that the record's key must be equal to the given record ID field.

### **FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.

### **GENERIC**

Specifies that the record ID field is generic, matching any with the given contents equal up the length of keylength.

### **GTEQ**

Specifies that the record's key must be greater than or equal to the given record ID field.

**KEYLENGTH(data-value)**

Specifies the length of the record ID key field.

**RBA**

Specifies that the key is by relative byte address.

**REQID(data-value)**

Specifies a request ID to be used in browsing operations.

**RIDFLD(data-area)**

Specifies the record ID field, the key used to search for the record. This data item is mapped to the column marked with an ridfld attribute in the XML file descriptor.

**RRN**

Specifies that the key is by relative record number.

**SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

**Conditions:**

FILENOTFOUN D	The file itself could not be found.
ILLOGIC	A logical error occurred.
INVREQ	Invalid request.
IOERR	An input/output error occurred, such as an invalid SQL query.
NOTAUTH	Not authorized.
NOTFND	The record was not found, such as an invalid record ID field.

## RETRIEVE

---

Retrieve data passed to the task by the START command.

If there are multiple retrieve records passed, each retrieve will fetch the next until ENDDATA is reached.

**Syntax:**

```
RETRIEVE
    {INTO(data-area) | SET(pointer-ref)}
```

[LENGTH(data-area)]  
[RTRANSID(data-area)]  
[RTERMID(data-area)]  
[QUEUE(data-area)]  
[WAIT]

### Service:

Interval Control

### Setup:

None

### Options:

#### **INTO(data-area)**

Retrieves the data passed by START FROM.

#### **SET(pointer-ref)**

Retrieves the data passed by START FROM as a pointer.

#### **LENGTH(data-area)**

Specifies the length of the INTO data-area.

Retrieves the length of the START FROM buffer actually passed.

#### **RTRANSID(data-area)**

Retrieves the 4-character RTRANSID passed by the START command.

#### **RTERMID(data-area)**

Retrieves the 4-character RTERMID passed by the START command.

#### **QUEUE(data-area)**

Retrieves the 8-character name of the temporary storage queue passed by START.

#### **WAIT**

### Conditions:

ENDDATA	ENDDATA is returned when the end of data passed is reached.
ENVDEFERR	Retrieve attempted to access a parameter that was not passed.
INVREQ	Invalid request.
LENGERR	The specified length is insufficient to hold the

	returned data.
NOTFND	

## RETURN

---

Return program control the next highest level, possibly setting the next transaction to process.

The ability to set the transaction is what allows pseudo-conversations to function. When a program has processed its input and generated its output, it marks the next transaction to receive the input from the current output.

### Syntax:

```

RETURN
    [TRANSID(name) [COMMAREA(data-area) [LENGTH(data-value)]]]
    [INPUTMSG(data-area) [INPUTMSGLEN(data-value)]]
    [ENDACTIVITY]

```

### Service:

Program Control

### Setup:

**pct.name=program\_uri**

program\_name must be a valid program reference, either a classname or URI.

**xlt.name=program\_name**

program\_name must be a valid name in the program table.

### Options:

**TRANSID(name)**

Returns the transaction name to use on the next leg of a pseudo-conversation, or to use immediately if IMMEDIATE is passed.

**COMMAREA(data-area)**

The COMMAREA to pass to the linked program.

**LENGTH(data-area)**

The length of the COMMAREA.

**INPUTMSG(data-area)**

Specifies data to be received by the other program's RECEIVE.

**INPUTMSGLEN(data-value)**

Specifies the length of the INPUTMSG.

**IMMEDIATE**

Specifies that the transaction is tied to the facility immediately.

**ENDACTIVITY**

This is used only for BTS (Business Transaction Services). This is ignored since BTS is not supported.

**Conditions:**

INVREQ	
1	Cannot return COMMAREA when no facility present.
2	IMMEDIATE or COMMAREA was passed at other than the top program level.
8	Cannot return INPUTMSG when no facility present.

LENGERR	
11	COMMAREA length not between 0 and 32767.
27	INPUTMSGLEN not between 0 and 32767

## REWIND COUNTER

---

Rewind the named counter that has reached its limit.

**Syntax:**

```
REWIND { COUNTER(name) | DCOUNTER(name) } [POOL(name)]  
      [INCREMENT(data-value)]
```

**Service:**

Counter

**Setup:**

**pool.name=redirected\_pool\_name**

If not specified, the pool name is the given name.

**counter.jndi=counter\_jndi\_name**

Specify the JNDI name of the Counter service. The default is 'CounterService'.

### Options:

#### **COUNTER(name)**

Specifies the name of the 32-bit counter.

#### **DCOUNTER(name)**

Specifies the name of the 64-bit counter.

#### **POOL(name)**

Specifies the name of the pool of counters. Each Counter service maintains its own pools of counters.

#### **INCREMENT(data-value)**

The counter is valid to reset only if already at its limit, or if the given increment would reach its limit.

### Conditions:

INVREQ	RESP2
201	Named counter invalid.
301	Server error.
303	Unexpected error, such as connectivity loss.
304	Invalid pool.
305	Cannot connect to server.
306	Serverabend.
308	This condition cannot occur (options table not loadable).
309	This condition cannot occur (options table error).
310	This condition cannot occur (options user exit).
403	The pool name contains invalid characters.
404	The counter name contains invalid characters.
406	The increment is invalid; it cannot be larger than the total range of the counter.

SUPPRESSED	RESP2
102	The counter has not reached its limit, including increment if specified.

## REWRITE

---

Rewrite a record previously read using the update option.

## Syntax:

```
REWRITE
    {FILE(data-value) | DATASET(data-value)}
    [TOKEN(data-area)]
    FROM(data-area)
    [SYSID(data-value) LENGTH(data-area) | LENGTH(data-area)]
    [NOSUSPEND]
```

## Service:

File Control

## Setup:

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**

The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

## Options:

### **DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

### **FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.

### **LENGTH(data-area)**

Specifies the length of the from field.

### **NOSUSPEND**

The read does not suspend on locks.

### **SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

### TOKEN(data-area)

Specifies a token relating this REWRITE back to a READ.

### Conditions:

FILENOTFOUND	The file itself could not be found.
ILLOGIC	A logical error occurred.
INVREQ	Invalid request.
IOERR	An input/output error occurred, such as an invalid SQL query.
LENGERR	The length of the record read exceeds the given INTO buffer's LENGTH.
NOTFND	The data record to rewrite was not found, such as having been just deleted.
NOTAUTH	Not authorized.
RECORDBUSY	The record was locked.

## ROUTE

---

(THIS COMMAND IS UNSUPPORTED)

### Syntax:

```
ROUTE
  [INTERVAL(hhmmss)]
  [AFTER
  [HOURS(data-value)]
  [MINUTES(data-value)]
  [SECONDS(data-value)]
  ]
  [TIME(hhmmss)]
  [AT
    [HOURS(data-value)]
    [MINUTES(data-value)]
    [SECONDS(data-value)]
  ]

  [ERRTERM | ERRTERM(data-value)]
  [TITLE(data-area)]
  [LIST(data-area)]
```

[OPCLASS(data-area)]

[REQID(data-value)]

[LDC(data-value)]

[NLEOM]

### Service:

Basic Mapping Support

### Setup:

None

### Options:

#### **INTERVAL(hhmmss)**

Specifies an interval of time duration data-value. The format is a single number with two-digits each hour, minute, second.

#### **TIME(hhmmss)**

Specifies a final time data-value. The format is a single number with two-digits each hour, minute, second.

#### **AFTER**

AFTER specifies that HOURS, MINUTES and SECONDS refers to an interval duration.

#### **AT**

AT specifies that HOURS, MINUTES and SECONDS refers to a final time.

#### **HOURS(data-value)**

Specifies the number of hours, and it must be within 0..99.

#### **MINUTES(data-value)**

Specifies the number of minutes, and it must be within 0..59 if other HOURS or SECONDS is specified.

#### **SECONDS(data-value)**

Specifies the number of seconds, and it must be within 0..59 if other HOURS or MINUTES is specified.

#### **FROM(data-area)**

Specifies the user data area to be passed to the new task.

#### **ERRTERM(data-value)**

Specifies a terminal to be notified of any error in routing the message.

#### **LDC(data-value)**

Specifies 2-character logical device code.

**LIST(data-area)**

Specifies list of terminals and operators.

**NLEOM**

New line end of message.

**OPCLASS(data-area)**

Specifies list of operator classes.

**REQID(data-value)**

Specifies 2-character prefix of temporary storage queue.

**TITLE(data-area)**

Specifies the title of the route message.

**Conditions:**

IGREQID
INVERRTERM
INVLDC
INVREQ
RTEFAIL
RTESOME

## SEND CONTROL

---

**Syntax:**

```
SEND CONTROL  
    [CURSOR | CURSOR(data-value)]  
    [FORMFEED]  
    [ERASE [DEFAULT | ALTERNATE] | ERASEAUP]  
    [PRINT]  
    [FREEKB]  
    [ALARM]  
    [FRSET]
```

**Service:**

Basic Mapping Support

### Setup:

None

### Options:

**CURSOR(data-value)**

**FORMFEED**

**ERASE**

**DEFAULT**

**ALTERNATE**

**ERASEAUP**

**PRINT**

**FREEKB**

**ALARM**

**FRSET**

### Conditions:

IGREQCD
IGREQID
INVLDC
INVPARTN
INVREQ
RETPAGE
TSIOERR
WRBRK

## SEND MAP

---

### Syntax:

```
SEND MAP(name)
    MAPSET(name)
    [FROM(data-area) | MAPONLY]
    [DATAONLY]
    [LENGTH(data-value)]
    [CURSOR | CURSOR(data-value)]
    [FORMFEED]
```

[ERASE [DEFAULT | ALTERNATE] | ERASEAUP]  
[PRINT]  
[FREEKB]  
[ALARM]  
[FRSET]

### Service:

Basic Mapping Support

### Setup:

None

### Options:

#### **MAP(name)**

The name is a literal data-value.

#### **MAPSET(name)**

The name is a literal data-value.

#### **FROM(data-area)**

#### **MAPONLY**

#### **DATAONLY**

#### **LENGTH(data-value)**

#### **CURSOR(data-value)**

#### **FORMFEED**

#### **ERASE**

#### **DEFAULT**

#### **ALTERNATE**

#### **ERASEAUP**

#### **PRINT**

#### **FREEKB**

#### **ALARM**

#### **FRSET**

# SEND PAGE

---

(THIS COMMAND IS UNSUPPORTED)

## Syntax:

```
SEND PAGE
    [RELEASE [TRANSID(name)] | RETAIN]
    [TRAILER(data-area)]
    [SET(pointer-ref)]
    [AUTOPAGE [CURRENT | ALL] | NOAUTOPAGE]
    [OPERPURGE]
    [FMHPARM(name)]
    [LAST]
```

## Service:

Basic Mapping Support

## Setup:

None

## Options:

**ALL**  
**AUTOPAGE**  
**CURRENT**  
**FMHPARM(name)**  
**LAST**  
**NOAUTOPAGE**  
**OPERPURGE**  
**RELEASE**  
**RETAIN**  
**SET(pointer-ref)**  
**TRAILER(data-area)**  
**TRANSID(name)**

**Conditions:**

IGREQCD
INVREQ
RETPAGE
TSIOERR
WRBRK

## SEND PARTNSET

---

(THIS COMMAND IS UNSUPPORTED)

**Syntax:**

SEND {PARTNSET | PARTNET(name)}

**Service:**

Basic Mapping Support

**Setup:**

None

**Options:**

**PARTNSET(name)**

**Conditions:**

INVPARTNSET
INVREQ

## SEND TEXT

---

**Syntax:**

```

SEND TEXT
    FROM(data-area)
    [LENGTH(data-value)]
    [CURSOR(data-value)]
    [FORMFEED]

```

[ERASE [DEFAULT|ALTERNATE]]  
[PRINT]  
[FREEKB]  
[ALARM]  
[NLEOM]  
[FMHPARM(name)]  
[LDC(name) | [OUTPARTN(name)] [ACTPARTN(name)]]  
[MSR(data-value)]

### Service:

Basic Mapping Support

### Setup:

None

### Options:

**FROM(data-area)**  
**LENGTH(data-value)**  
**CURSOR(data-value)**  
**FORMFEED**  
**ERASE**  
**DEFAULT**  
**ALTERNATE**  
**PRINT**  
**FREEKB**  
**ALARM**  
**NLEOM**  
**FMHPARM(name)**  
**LDC(name)**  
**OUTPARTN(name)**  
**ACTPARTN(name)**  
**MSR(data-value)**

**Conditions:**

IGREQCD
IGREQID
INVLDC
INVPARTN
INVREQ
LENGERR
RETPAGE
TSIOERR
WRBRK

## SIGNOFF

---

This command is dependent upon the capabilities of the External Security Manager.

**Syntax:**

SIGNOFF

**Service:**

Security

**Setup:**

None

**Options:**

None

**Conditions:**

INVREQ	Invalid request.
--------	------------------

## SIGNON

---

This command is dependent upon the capabilities of the External Security Manager.

## Syntax:

SIGNON

USERID(data-value)

[ESMREASON(data-area)]

[ESMRESP(data-area)]

[GROUPLD(data-value)]

[LANGUAGECODE(data-value) | NATLANG(data-value)]

[LANGINUSE(data-area)]

[NATLANGINUSE(data-area)]

[PASSWORD(data-value)]

[NEWPASSWORD(data-value)]

[OIDCARD(data-value)]

## Service:

Security

## Setup:

None

## Options:

### **ESMREASON(data-area)**

Retrieves the external security manager reason code.

### **ESMRESP(data-area)**

Retrieves the external security manager response code.

### **GROUPLD(data-value)**

Specifies the group ID for a user.

### **LANGUAGECODE(data-value)**

Specifies the 3-character language code.

### **LANGINUSE(data-area)**

Retrieves the 3-character language code assigned by login.

### **NATLANG(data-value)**

Specifies the 1-character language code.

### **NATLANGINUSE(data-area)**

Retrieves the 1-character language code assigned by login.

**NEWPASSWORD(data-value)**

Specifies the new user password.

**OIDCARD(data-value)**

Specifies a 65-byte magnetic strip value.

**PASSWORD(data-value)**

Specifies the current user password.

**USERID(data-value)**

Specifies the user ID.

**Conditions:**

INVREQ	Invalid request.
NOTAUTH	Not authorized.
USERIDERR	The user ID is invalid.

## SPOOLCLOSE

---

Close the spool file.

**Syntax:**

```
SPOOLCLOSE TOKEN(data-area)
      [ KEEP | DELETE ]
```

**Service:**

Spool

**Setup:**

A spool queue for output is defined as:

```
spool[.userid[.class[.node]]]=uri
```

and a spool queue for input as:

```
spool[.userid[.class]]=uri
```

**Options:****TOKEN(data-area)**

The 8-character token contents determine the spool file to be closed. It must be one opened previously by SPOOLOPEN.

## KEEP

Specifies disposition of the closing spool file.

For input, the same spool file will be read at the next open input.

For output, the spool file is to be passed to its destination node.

## DELETE

Specifies disposition of the closing spool file.

For input, the next open is the next spool file will be opened at the next open input.

For output, the spool file is to be purged.

### Conditions:

NOSPOOL	
4	The specified spool file could not be found. The token may be invalid.

NOTOPEN	
8	The spool file is not open, and so could not be closed.

## SPOOL OPEN INPUT

---

Open a spool file for input.

### Syntax:

```
SPOOL OPEN INPUT TOKEN(data-area)
                USERID(data-value)
                [CLASS(data-value)]
```

### Service:

Spool

### Setup:

```
spool[.userid[.class]]=uri
```

### Options:

#### TOKEN(data-area)

The 8-character token contents determine the spool file to be closed. It must be one opened previously by SPOOL OPEN.

**USERID(data-value)**

Specifies the user id to be associated with the spool file.

**CLASS(data-value)**

Specifies a one (1) character class designation.

**Conditions:**

NOSPOOL	
4	The specified spool file could not be found. The token may be invalid.

INVREQ	
16	USERID not specified.
36	Neither INPUT nor OUTPUT was specified.

ILLOGIC	
3	Invalid CLASS value.

## SPOOL OPEN OUTPUT

---

Open a spool file for output.

**Syntax:**

```

SPOOL OPEN OUTPUT TOKEN(data-area)
    USERID(data-value)
    NODE(data-value)
    [CLASS(data-value)]
    OUTDESCR(pointer-ref)
    [NOCC | ASA | MCC]
    [PRINT | PRINT [RECORDLENGTH(data-value)] | PUNCH]
  
```

**Service:**

Spool

**Setup:**

```
spool[.userid[.class[.node]]]=uri
```

## Options:

### **TOKEN(data-area)**

The 8-character token contents determine the spool file to be closed. It must be one opened previously by SPOOLOPEN.

### **USERID(data-value)**

Specifies the user id to be associated with the spool file.

### **CLASS(data-value)**

Specifies a one (1) character class designation.

### **NODE(data-value)**

Specifies the eight (8) character destination node.

### **OUTDESCR(pointer-ref)**

The OUTDESCR option is not supported, and will raised an UNSUPPORTED\_OPTION condition if used

### **NOCC**

Requests that the records in the spool file do not include internal carriage controls.

### **ASA**

Requests that records in the spool file begin with an ASA carriage-control character.

### **MCC**

Requests that records in the spool file begin with an IBM Machine Command Code character.

### **PRINT**

The default, specifies that large records (up to 32760 bytes) may be sent to the spool.

### **RECORDLENGTH(data-value)**

Specifies the maximum length of a record in the spool file. The default is 32760.

### **PUNCH**

Specifies that the record length is 80.

## Conditions:

NOSPOOL	
4	The specified spool file could not be found. The token may be invalid.

INVREQ	
--------	--

16	USERID not specified.
20	NODE not specified.
36	Neither INPUT nor OUTPUT was specified.

ILLOGIC	
3	Invalid CLASS value.

LENGERR	RECORDLENGTH was invalid, not between 0 and 32760. RESP2 contains the invalid valid.
---------	--

UNSUPPORTED_OPTION	OUTDESCR is not supported.
--------------------	----------------------------

## SPOOLREAD

---

Read data from a spool file opened for input.

### Syntax:

```
SPOOLREAD TOKEN(data-area)
        INTO(data-area)
        [MAXLENGTH(data-value)]
        [TOLENGTH(data-area)]
```

### Service:

Spool

### Setup:

```
spool[.userid[.class]]=uri
```

### Options:

#### **TOKEN(data-area)**

The 8-character token contents determine the spool file to be closed. It must be one opened previously by SPOOLOPEN.

#### **INTO(data-area)**

Specifies the data area to which the read record is copied.

#### **MAXLENGTH(data-value)**

Specifies the maximum length of the input record.

### **TOFLENGTH(data-area)**

Returns the length of the data read.

#### **Conditions:**

ENDFILE	The spool file has been completely read.
ILLOGIC	
3	The CLASS is invalid.
INVREQ	
24	The INTO is not specified
NOSPOOL	
4	The specified spool file could not be found. The token may be invalid.
NOTOPEN	
8	The spool file is not open.
12	The spool file was opened for output.

## **SPOOLWRITE**

---

Write data to a spool file opened for output.

#### **Syntax:**

```
SPOOLWRITE TOKEN(data-area)
      FROM(data-area)
      [MAXFLENGTH(data-value)]
      [FLENGTH(data-value)]
      [LINE | PAGE]
```

#### **Service:**

Spool

#### **Setup:**

```
spool[.userid[.class[.node]]]=uri
```

## Options:

### **TOKEN(data-area)**

The 8-character token contents determine the spool file to be closed. It must be one opened previously by SPOOLOPEN.

### **FROM(data-value)**

Specifies the data record to write to the spool file.

### **FLENGTH(data-area)**

Specifies the length of the data record to write to the spool file. The length must be between 1 and the record length specified on the open.

### **LINE**

Specifies LINE format.

### **PAGE**

Specifies PAGE format.

## Conditions:

ENDFILE	The spool file has been completely read.
ILLOGIC	
3	The CLASS is invalid.
INVREQ	
28	The FROM is not specified.
NOSPOOL	
4	The specified spool file could not be found. The token may be invalid.
NOTOPEN	
8	The spool file is not open.
16	The spool file was opened for input.
LENGERR	The length is not between 1 and the opened record length.

## START

---

Start a separate task at a specific time.

The separate task is executed as a JEE message bean. It has no terminal and executed in the background.

INTERVAL or AFTER specifies a period of time, whereas TIME or AT specifies the final time itself.

### Syntax:

```
START
    [TRANSID(data-value)]
    [REQID(data-value)]

    [INTERVAL(hhmmss)]
    [AFTER
        [HOURS(data-value)]
        [MINUTES(data-value)]
        [SECONDS(data-value)]
    ]
    [TIME(hhmmss)]
    [AT
        [HOURS(data-value)]
        [MINUTES(data-value)]
        [SECONDS(data-value)]
    ]
    [FROM(data-area) [LENGTH(data-value) [FMH]]]
    [USERID(data-value) | TERMID(data-value)]
    [SYSID(data-value)]
    [RTRANSID(data-value)]
    [RTERMID(data-value)]
    [QUEUE(data-value)]
    [NOCHECK]
    [PROTECT]
```

### Service:

Interval Control

### Setup:

```
queue[.<sysid>].etpsysq=jms:<message_queue_jndi_name>
```

The default Elastic Transaction Platform System Queue is etpsysq\_<sysid>. Each deployment has its own system message queue.

```
<message_queue_jndi_name>.factory=<jms_factory_jndi_name>
```

If not present, then the `jms.default.factory` setting is used.

`<message_queue_jndi_name>.name=<jms_login_user_name>`

`<message_queue_jndi_name>.password=<jms_login_password>`

The user and password are only required if JMS is setup to require them.

Additionally, the transaction must be setup the same as for link wherever the start message will be received. The transaction property `restart` may be set to allow the transaction to be automatically restarted upon failure.

## Options:

### **TRANSID(data-value)**

Specifies the transaction ID to execute.

### **REQID(data-value)**

Specifies a request ID through which this command may be cancelled using the `CANCEL` command. The request ID is valid only within the same session.

### **INTERVAL(hhmmss)**

Specifies an interval of time duration `data-value`. The format is a single number with two-digits each hour, minute, second.

### **TIME(hhmmss)**

Specifies a final time `data-value`. The format is a single number with two-digits each hour, minute, second.

### **AFTER**

`AFTER` specifies that `HOURS`, `MINUTES` and `SECONDS` refers to an interval duration.

### **AT**

`AT` specifies that `HOURS`, `MINUTES` and `SECONDS` refers to a final time.

### **HOURS(data-value)**

Specifies the number of hours, and it must be within 0..99.

### **MINUTES(data-value)**

Specifies the number of minutes, and it must be within 0..59 if other `HOURS` or `SECONDS` is specified.

### **SECONDS(data-value)**

Specifies the number of seconds, and it must be within 0..59 if other `HOURS` or `MINUTES` is specified.

### **FROM(data-area)**

Specifies the user data area to be passed to the new task.

### **LENGTH(data-value)**

Specifies the length of the user data area to be passed to the new task.

**FMH**

Specifies that the user data contains function management headers.

**USERID(data-value)**

Specifies the user ID to be used for the new task.

**TERMID(data-value)**

Specifies the terminal ID for the new task. As all terminals are virtual bridges, and the new task may not have a terminal, this option is not supported. Any virtual terminal must activate its own task. The way to associate a terminal with a timed task is to have the client virtual terminal poll a transaction at the specified time, with all such client-driven interaction controlled by the client.

**SYSID(data-value)**

Specifies the system ID for the new task. This is used in finding the correct system message queue.

**RTRANSID(data-value)**

Specifies a 4-character name passed to the new task, traditionally a transaction ID.

**RTERMID(data-value)**

Specifies a 4-character name passed to the new task, traditionally a terminal ID.

**QUEUE(data-value)**

Specifies an 8-character name passed to the new task, traditionally a temporary storage queue name.

**NOCHECK**

Specifies that less error checking will occur for possibly improved performance. This option is ignored.

**PROTECT**

Specifies that a syncpoint must occur before the START may take place.

**Conditions:**

TRANSIDERR	The transaction ID is invalid.
LENGERR	The length is less than zero (0).
INVREQ	RESP2
	4 Hours are invalid. 5 Minutes are invalid. 6 Seconds are invalid.
QIDERR	The system message queue used to pass start messages has failed.

IOERR	Communication with the system message queue used to pass start messages has failed.
-------	---

## SUSPEND

---

Suspend a task, allowing other tasks to proceed first where the task scheduler permits. This is otherwise known as a yield.

### Syntax:

SUSPEND

### Service:

Task Control

### Options:

None

## START ATTACH

---

Start a separate task immediately.

The separate task is executed as a JEE message bean. It has no terminal and executed in the background.

INTERVAL or AFTER specifies a period of time, whereas TIME or AT specifies the final time itself.

### Syntax:

```
START ATTACH
    [TRANSID(data-value)]
    [FROM(data-area) [LENGTH(data-value)]]
```

### Service:

Interval Control

### Setup:

**queue[.<sysid>].etpsysq=jms:<message\_queue\_jndi\_name>**

The default Elastic Transaction Platform System Queue is etpsysq\_<sysid>. Each deployment has its own system message queue.

**<message\_queue\_jndi\_name>.factory=<jms\_factory\_jndi\_name>**

If not present, then the jms.default.factory setting is used.

**<message\_queue\_jndi\_name>.name=<jms\_login\_user\_name>**

**<message\_queue\_jndi\_name>.password=<jms\_login\_password>**

The user and password are only required if JMS is setup to require them.

Additionally, the transaction must be setup the same as for link wherever the start message will be received. The transaction property restart may be set to allow the transaction to be automatically restarted upon failure.

### Options:

#### **TRANSID(data-value)**

Specifies the transaction ID to execute.

#### **REQID(data-value)**

Specifies a request ID through which this command may be cancelled using the CANCEL command. The request ID is valid only within the same session.

### Conditions:

EXPIRED	The interval has already expired.
INVREQ	RESP2
	4 Hours are invalid. 5 Minutes are invalid. 6 Seconds are invalid.

## START BREXIT

---

(THIS COMMAND IS UNSUPPORTED)

The START BREXIT command is not supported. The Elastic Transaction Platform always operates under an externally defined bridge.

### Syntax:

START

{BREXIT | BREXIT(data-value)}

TRANSID(data-value)

[BRDATA(data-area) [BRDATALENGTH(data-value)]]

[USERID(data-value)]

## Service:

Interval Control

## Options:

**BREXIT(data-value)**

**TRANSID(data-value)**

**BRDATA(data-area)**

**BRDATALENGTH(data-value)**

**USERID(data-value)**

# STARTBR

---

Start browsing a file, preparing it for use with READNEXT or READPREV. The browse must be terminated using ENDBR when finished.

Currently, the only records available during a browse operation are those which are in the scope of the STARTBR operation. A READPREV cannot go further back than STARTBR's browse implies. A READNEXT cannot go further forward than STARTBR's browse implies.

## Syntax:

```
STARTBR
    {FILE(data-value) | DATASET(data-value)}
    RIDFLD(data-area) [KEYLENGTH(data-value) [GENERIC]]
    [REQID(data-value)]
    [SYSID(data-value)]
    [RBA | RRN | DEBKEY | DEBREC]
    [GTEQ | EQUAL]
```

## Service:

File Control

## Setup:

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**

The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

## Options:

### **DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

### **DEBKEY**

Deblocking by key in BDAM, this option is not used.

### **DEBREC**

Deblocking by record in BDAM, this option is not used.

### **EQUAL**

Specifies that the record's key must be equal to the given record ID field.

### **FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.

### **GENERIC**

Specifies that the record ID field is generic, matching any with the given contents equal up the length of keylength.

### **GTEQ**

Specifies that the record's key must be greater than or equal to the given record ID field.

### **KEYLENGTH(data-value)**

Specifies the length of the record ID key field.

### **RBA**

Specifies that the key is by relative byte address.

### **REQID(data-value)**

Specifies a request ID to be used in browsing operations.

### **RIDFLD(data-area)**

Specifies the record ID field, the key used to search for the record. This data item is mapped to the column marked with an ridfld attribute in the XML file descriptor.

### **RRN**

Specifies that the key is by relative record number.

### **SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

**Conditions:**

FILENOTFOUND	The file itself could not be found.
ILLOGIC	A logical error occurred.
INVREQ	Invalid request.
IOERR	An input/output error occurred, such as an invalid SQL query.
NOTAUTH	Not authorized.
NOTFND	The record was not found, such as an invalid record ID field.
NOTOPEN	The file is not open.

## SUSPEND

---

Suspend a task, allowing other tasks to proceed first where the task scheduler permits. This is otherwise known as a yield.

**Syntax:**

SUSPEND

**Service:**

Task Control

**Setup:**

None

**Options:**

None

**Conditions:**

None

# SYNCPPOINT

---

Commit a unit of work within a task. A commit occurs automatically when a task ends normally.

**Syntax:**

SYNCPPOINT

**Service:**

Recovery

**Options:**

None

**Conditions:**

ROLLBACK	Occurs when the task cannot commit, but rather is forced to rollback. Generally, this will occur when a remotely linked program must rollback.
----------	--

# SYNCPPOINT ROLLBACK

---

Rollback a unit of work within a task. A rollback occurs automatically when a task ends abnormally.

EIBSYNRB, EIBERR and EIBERRCD may be set.

**Syntax:**

SYNCPPOINT ROLLBACK

**Service:**

Recovery

**Options:**

None

### Conditions:

INVREQ	RESP2
200	SYNCPOINT ROLLBACK failed.

## TRACE

---

Enable or disable tracing facilities. The trace command affects only the environment in which it is executed, not the global environment.

### Syntax:

```
TRACE {ON | OFF}
      [SYSTEM]
      [EI]
      [USER]
      [SINGLE]
```

### Setup:

**trace.master={true|false}**

The trace.master switch enables or disables all physical tracing. If trace.master=false, then the tracing commands function but no actual output is created.

**trace=journal\_name**

The tracing destination is a journal name, which in turn is specified through journal.name=journal\_protocol.

### Service:

TraceControl

### Options:

**ON**

TRACE settings are enabled.

**OFF**

TRACE settings are disabled.

**SYSTEM**

TRACE setting applies to all tracing system activity.

**EI**

TRACE settings applies to EXEC interface activity.

### **USER**

TRACE setting applies to user entries.

### **SINGLE**

TRACE setting applies only to user trace entries within the current task. All USER entries are only within the current task, not global, so this is no different than USER.

## **UNLOCK**

---

Release file locks. As SQL uses a different mechanism for concurrency control, this command does nothing.

### **Syntax:**

```
UNLOCK
      {FILE(data-value) | DATASET(data-value)}
      [TOKEN(data-area)]
      [SYSID(data-value)]
```

### **Service:**

File Control

### **Setup:**

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**

The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

### **Options:**

#### **DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

#### **FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.

**SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

**TOKEN(data-area)**

Specifies a token relating this REWRITE back to a READ.

**Conditions:**

None

## UPDATE COUNTER

---

Update the current value of the named counter.

**Syntax:**

```
UPDATE { COUNTER(name) | DCOUNTER(name) }  
      [POOL(name)]  
      VALUE(data-value)  
      [COMPAREMIN(data-value)]  
      [COMPAREMAX(data-value)]
```

**Service:**

Counter

**Setup:**

**pool.name=redirected\_pool\_name**

If not specified, the pool name is the given name.

**counter.jndi=counter\_jndi\_name**

Specify the JNDI name of the Counter service. The default is 'CounterService'.

**Options:**

**COUNTER(name)**

Specifies the name of the 32-bit counter.

**DCOUNTER(name)**

Specifies the name of the 64-bit counter.

**POOL(name)**

Specifies the name of the pool of counters. Each Counter service maintains its own pools of counters.

**VALUE(data-value)**

Specifies the value of the counter.

**COMPAREMIN(data-value)**

Specifies a minimum value against which to compare the value. If the value is greater than or equal, then no action is taken; if less than the minimum, a condition is raised.

If COMPAREMIN>COMPAREMAX, then either condition must be true rather than both.

Success of the command is conditional upon the comparison.

**COMPAREMAX(data-value)**

Specifies a maximum value against which to compare the value. If the value is less than or equal, then no action is taken; if less than the minimum, a condition is raised.

If COMPAREMIN>COMPAREMAX, then either condition must be true rather than both.

Success of the command is conditional upon the comparison.

**Conditions:**

INVREQ	RESP2
201	Named counter invalid.
301	Server error.
303	Unexpected error, such as connectivity loss.
304	Invalid pool.
305	Cannot connect to server.
306	Server abend.
308	This condition cannot occur (options table not loadable).
309	This condition cannot occur (options table error).
310	This condition cannot occur (options user exit).
403	The pool name contains invalid characters.
404	The counter name contains invalid characters.
406	The increment is invalid; it cannot be larger than the total range of the counter.

SUPPRESSED	RESP2
103	The value is not within COMPAREMIN and

	COMPAREMAX, or beyond the limits if only one of COMPAREMIN and COMPAREMAX is used.
--	--

# UNLOCK

---

Release file locks. As SQL uses a different mechanism for concurrency control, this command does nothing.

**Syntax:**

```
UNLOCK
      {FILE(data-value) | DATASET(data-value)}
      [TOKEN(data-area)]
      [SYSID(data-value)]
```

**Service:**

File Control

**Setup:**

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**  
 The xml\_file\_descriptor is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a \$XFD FILE="filename.xml" command before a record in the data division will generate a starting template in the listing folder. The sql\_connection\_name is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**  
 The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

**Options:**

- DATASET(data-value)**  
 Specifies the filename the same as FILE, but FILE is preferred.
- FILE(data-value)**  
 Specifies the filename. This filename is referred to in the setup via file.<filename>. For example, FILE('abc') would refer to setup entry 'file.abc=jdbc:abc.xml'.
- SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

**TOKEN(data-area)**

Specifies a token relating this REWRITE back to a READ.

**Conditions:**

None

## **VERIFY PASSWORD**

---

This command is dependent upon the capabilities of the External Security Manager.

**Syntax:**

```
VERIFY PASSWORD(data-value)
    USERID(data-value)
    [CHANGETIME(data-area)]
    [DAYSLEFT(data-area)]
    [ESMREASON(data-area)]
    [ESMRESP(data-area)]
    [EXPIRYTIME(data-area)]
    [INVALIDCOUNT(data-area)]
    [LASTUSETIME(data-area)]
```

**Service:**

Security

**Setup:**

None

**Options:**

**PASSWORD(data-value)**

Specifies the user's password.

**USERID(data-value)**

Specifies the user id.

**CHANGETIME(data-area)**

Retrieves data and time password last changed, as in ABSTIME.

**DAYSLEFT(data-area)**

Retrieves number of days until password expires, -1 if no expiration.

**ESMREASON(data-area)**

Retrieves the external security manager reason code.

**ESMRESP(data-area)**

Retrieves the external security manager response code.

**EXPIRYTIME(data-area)**

Retrieves date and time password expires, as in ABSTIME.

**INVALIDCOUNT(data-area)**

Retrieves count of invalid password attempts by user.

**LASTUSETIME(data-area)**

Retrieves last time user id was used, as in ABSTIME.

**Conditions:**

INVREQ	Invalid request.
NOTAUTH	Not authorized.
USERID	Invalid user ID.

## WAIT EXTERNAL

---

(THIS COMMAND IS UNSUPPORTED)

This command is not supported and not intended for COBOL usage.

Synchronize events.

**Syntax:**

```

WAIT EXTERNAL
    ECBLIST(pointer-value)
    NUMEVENTS(data-value)
    {PURGEABLE | PURGEABILITY(cvda) | NOTPUREGABLE}
    [NAME(data-value)]

```

**Service:**

Task Management

**Setup:**

None

**Options:**

None

**Conditions:**

None

## WAIT EVENT

---

Wait for a post event to occur.

**Syntax:**

```
WAIT EVENT
      ECADDR(pointer-value)
      [NAME(data-value)]
```

**Service:**

Interval Control

**Setup:**

None

**Options:**

**ECADDR(pointer-value)**

Specifies the pointer returned via the POST command. This pointer-value cannot be another source, or the post will wait indefinitely.

**NAME(data-value)**

Specifies a name for the wait. This value is not currently used.

**Conditions:**

EXPIRED	The interval has already expired.
INVREQ	RESP2

	4 Hours are invalid. 5 Minutes are invalid. 6 Seconds are invalid.
--	--

## WAIT JOURNALNAME

---

### Syntax:

```
WAIT JOURNALNAME(data-value)
      [REQID(data-value)]
```

### Service:

Journal

### Setup:

journal.<name>=<journal\_protocol\_reference>

### Options:

#### JOURNALNAME(data-value)

Data-value is a two-digit binary value, representing the journal name DFHJnn, where nn is the two-digit value.

#### REQID(data-value)

Specify the request ID obtained from WRITE JOURNALNAME on which to wait. Note that many journal protocols will wait on the entire journal to be synchronized rather than only a particular request.

### Conditions:

INVREQ	There have been no writes on which to wait.
IOERR	Input/output error.
JIDERR	The journal reference is unknown.
NOTOPEN	The journal itself was recognized but could not be opened.

## WAIT JOURNALNUM

---

Wait for the journal to be completely written before continuing.

### Syntax:

```
WAIT JOURNALNUM(data-value)
      [REQID(data-value)]
```

### Service:

Journal

### Setup:

```
journal.DFHJ<nn>=<journal_protocol_reference>
```

### Options:

#### **JOURNALNUM(data-value)**

Data-value is a two-digit binary value, representing the journal name DFHJnn, where nn is the two-digit value.

#### **REQID(data-value)**

Specify the request ID obtained from WRITE JOURNALNUM on which to wait. Note that many journal protocols will wait on the entire journal to be synchronized rather than only a particular request.

### Conditions:

INVREQ	There have been no writes on which to wait.
IOERR	Input/output error.
JIDERR	The journal reference is unknown.
NOTOPEN	The journal itself was recognized but could not be opened.

## WAITCICS

---

(THIS COMMAND IS UNSUPPORTED)

This command is not supported and not intended for COBOL usage.

### Syntax:

```
WAITCICS
      ECBLIST(pointer-value)
      NUMEVENTS(data-value)
      {PURGEABLE | PURGEABILITY(cvda) | NOTPURGEABLE}
```

[NAME(data-value)]

**Service:**

Task Management

**Setup:**

None

**Options:**

None

**Conditions:**

None

## WAIT JOURNALNAME

---

**Syntax:**

WAIT JOURNALNAME(data-value)  
[REQID(data-value)]

**Service:**

Journal

**Options:**

**JOURNALNAME(data-value)**

Data-value is a two-digit binary value, representing the journal name DFHJnn, where nn is the two-digit value.

**REQID(data-value)**

Specify the request ID obtained from WRITE JOURNALNAME on which to wait. Note that many journal protocols will wait on the entire journal to be synchronized rather than only a particular request.

**Conditions:**

INVREQ	There have been no writes on which to wait.
--------	---

IOERR	Input/output error.
JIDERR	The journal reference is unknown.
NOTOPEN	The journal itself was recognized but could not be opened.

## WAIT JOURNALNUM

---

Wait for the journal to be completely written before continuing.

### Syntax:

```
WAIT JOURNALNUM(data-value)
      [REQID(data-value)]
```

### Service:

Journal

### Options:

#### **JOURNALNUM(data-value)**

Data-value is a two-digit binary value, representing the journal name DFHJnn, where nn is the two-digit value.

#### **REQID(data-value)**

Specify the request ID obtained from WRITE JOURNALNUM on which to wait. Note that many journal protocols will wait on the entire journal to be synchronized rather than only a particular request.

### Conditions:

INVREQ	There have been no writes on which to wait.
IOERR	Input/output error.
JIDERR	The journal reference is unknown.
NOTOPEN	The journal itself was recognized but could not be opened.

## WEB ENDBROWSE COOKIE

---

End the browsing of HTTP cookies.

### Syntax:

WEB ENDBROWSE COOKIE

### Service:

Web

### Options:

None

### Conditions:

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.
4	There was no prior WEB STARTBROWSE COOKIE.

## WEB ENDBROWSE FORMFIELD

---

End the browsing of HTTP form fields.

### Syntax:

WEB ENDBROWSE FORMFIELD

### Service:

Web

### Options:

None

### Conditions:

INVREQ	
1	The command is a not a Web application.

3	There is no HTTP request.
4	There was no prior WEB STARTBROWSE FORMFIELD.

## WEB ENDBROWSE HTTPHEADER

---

End the browsing of HTTP headers.

### Syntax:

WEB ENDBROWSE HTTPHEADER

### Service:

Web

### Options:

None

### Conditions:

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.
4	There was no prior WEB STARTBROWSE HTTPHEADER.

## WEB EXTRACT

---

Extract information about the Web HTTP request.

Note that not all fields are available from all HTTP requests.

### Syntax:

WEB EXTRACT

[HTTPMETHOD(data-area) METHODLENGTH(data-area)]

[HTTPVERSION(data-area) VERSIONLEN(data-area)]

[PATH(data-area) PATHLENGTH(data-area)]

[REQUESTTYPE(cvda)]

[AUTHTYPE(data-area) AUTHLENGTH(data-area)]

[CONTEXTPATH(data-area) CONTEXTPATHLENGTH(data-area)]  
 [PATHTRANSLATED(data-area) PATHTRANSLATEDLENGTH(data-area)]  
 [QUERYSTRING(data-area) QUERYSTRINGLENGTH(data-area)]  
 [REMOTEUSER(data-area) REMOTEUSERLENGTH(data-area)]  
 [REQUESTEDSESSIONID(data-area) REQUESTEDSESSIONIDLENGTH(data-area)]  
 [REQUESTURI(data-area) REQUESTURILENGTH(data-area)]  
 [REQUESTURL(data-area) REQUESTURLLENGTH(data-area)]  
 [SERVLETPATH(data-area) SERVLETPATHLENGTH(data-area)]  
 [CHARACTERENCODING(data-area) CHARACTERENCODINGLENGTH(data-area)]  
 [CONTENTLENGTH(data-area)]  
 [CONTENTTYPE(data-area) CONTENTTYPELENGTH(data-area)]  
 [LOCALE(data-area) LOCALELENGTH(data-area)]  
 [PROTOCOL(data-area) PROTOCOLLENGTH(data-area)]  
 [REMOTEADDR(data-area) REMOTEADDRLENGTH(data-area)]  
 [REMOTEHOST(data-area) REMOTEHOSTLENGTH(data-area)]  
 [SCHEME(data-area) SCHEMELENGTH(data-area)]  
 [SERVERNAME(data-area) SERVERNAMELENGTH(data-area)]  
 [SERVERPORT(data-area)]  
 [SECURE(cvda)]

### Service:

Web

### Options:

#### **HTTPMETHOD(data-area)**

Returns the HTTP method (e.g. "GET").

#### **METHODLENGTH(data-area)**

Pass HTTPMETHOD buffer length, returns the length of the extracted HTTPMETHOD.

#### **HTTPVERSION(data-area)**

Returns the HTTP version string (e.g. "HTTP/1.1").

#### **VERSIONLEN(data-area)**

Pass HTTPVERSION buffer length, returns the length of the extracted HTTPVERSION.

**PATH(data-area)**

Returns the HTTP path string.

**PATHLENGTH(data-area)**

Pass PATH buffer length, returns the length of the extracted PATH.

**REQUESTTYPE(cvda)**

Returns HTTP or NONHTTP

**AUTHTYPE(data-area)****AUTHLENGTH(data-area)****CONTEXTPATH(data-area)**

(e.g. "/myapp").

**CONTEXTPATHLENGTH(data-area)****PATHTRANSLATED(data-area)****PATHTRANSLATEDLENGTH(data-area)****QUERYSTRING(data-area)**

Returns the query string from the URL, the portion after the question mark (?). (e.g. "hl=en&ie=UTF-8&oe=UTF-8&q=Search+Query").

**QUERYSTRINGLENGTH(data-area)****REMOTEUSER(data-area)**

Returns the remote user if logged in (e.g. "root").

**REMOTEUSERLENGTH(data-area)****REQUESTEDSESSIONID(data-area)****REQUESTEDSESSIONIDLENGTH(data-area)****REQUESTURI(data-area)**

(e.g. "/myapp/myapp").

**REQUESTURILENGTH(data-area)****REQUESTURL(data-area)**

Returns the entire URL for this web request. (e.g. "<http://192.168.0.234:7001/myapp/myapp>").

**REQUESTURLLENGTH(data-area)****SERVLETPATH(data-area)**

(e.g. "/myapp").

**SERVLETPATHLENGTH(data-area)****CHARACTERENCODING(data-area)**

Returns the character encoding.

**CHARACTERENCODINGLENGTH(data-area)**

Returns length of CHARACTERENCODING.

**CONTENTLENGTH(data-area)**

Returns the content length.

**CONTENTTYPE(data-area)****CONTENTTYPELENGTH(data-area)****LOCALE(data-area)**

Returns the locale in language2\_COUNTRY2 format. (e.g. "en\_US").

**LOCALELENGTH(data-area)****PROTOCOL(data-area)**

(e.g. "HTTP/1.1").

**PROTOCOLLENGTH(data-area)****REMOTEADDR(data-area)**

Returns the remote machine's TCP/IP address. (e.g. "192.168.0.123").

**REMOTEADDRLENGTH(data-area)****REMOTEHOST(data-area)**

Returns the remote machine's hostname, if possible. (e.g. "mycomputer.internal.myhost.com").

**REMOTEHOSTLENGTH(data-area)****SCHEME(data-area)**

(e.g. "http").

**SCHEMELENGTH(data-area)****SERVERNAME(data-area)**

(e.g. "192.168.0.234").

**SERVERNAMELENGTH(data-area)****SERVERPORT(data-area)**

Returns the server TCP/IP port.

**SECURE(cvda)**

Returns SSL or NOSSL.

**Conditions:**

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.
LENGERR	

1	The passed LENGTH was less than zero (0).
---	---

## WEB READ COOKIE

---

Retrieve named HTTP cookie.

### Syntax:

WEB READ

COOKIE(data-area)

NAMELENGTH(data-value)

VALUE(data-area)

VALUELENGTH(data-area)

COMMENT(data-area) COMMENTLENGTH(data-area)

DOMAIN(data-area) DOMAINLENGTH(data-area)

MAXAGE(data-area)

PATH(data-area) PATHLENGTH(data-area)

SECURE(cvda)

VERSION(data-area)

### Service:

Web

### Options:

#### **COOKIE(data-area)**

Specify the HTTP cookie to retrieve.

#### **NAMELENGTH(data-value)**

Specify the length of the HTTP cookie.

#### **VALUE(data-area)**

Retrieves the value of the HTTP cookie.

#### **VALUELENGTH(data-area)**

Pass the length of the VALUE buffer, retrieves the length of the VALUE.

#### **COMMENT(data-area)**

Retrieves the cookie's comment. (Comment is not supported by early versions of JEE.)

#### **COMMENTLENGTH(data-area)**

Pass the length of the COMMENT buffer, retrieve the length of the retrieved comment.

**DOMAIN(data-area)**

Retrieves the cookie's domain.

**DOMAINLENGTH(data-area)**

Pass the length of the DOMAIN buffer, retrieve the length of the retrieved domain.

**MAXAGE(data-area)**

Retrieves the maximum age of the cookie.

**PATH(data-area)**

Retrieves the path of the cookie.

**PATHLENGTH(data-area)**

Pass the length of the PATH buffer, retrieve the length of the retrieved path.

**SECURE(data-area)**

Returns SECURE or NOSECURE, reflecting the status of cookie.

**VERSION(data-area)**

Returns the version of the cookie.

**Conditions:**

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

LENGERR	
1	The passed VALUELENGTH was less than zero (0).
2	The cookie value was truncated, buffer too small.

NOTFND	
1	The cookie could not be found.

## **WEB READ FORMFIELD**

---

Retrieve named HTTP form field.

## Syntax:

WEB READ  
FORMFIELD(data-area)  
NAMELENGTH(data-value)  
VALUE(data-area)  
VALUELENGTH(data-area)

## Service:

Web

## Options:

### **FORMFIELD(data-area)**

Specify the HTTP form field to retrieve.

### **NAMELENGTH(data-value)**

Specify the length of the HTTP form field.

### **VALUE(data-area)**

Retrieves the value of the HTTP form field.

### **VALUELENGTH(data-area)**

Pass the length of the VALUE buffer, retrieves the length of the VALUE.

## Conditions:

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

LENGERR	
1	The passed VALUELENGTH was less than zero (0).
2	The form field value was truncated, buffer too small.

NOTFND	
1	The cookie could not be found.

# WEB READ HTTPHEADER

---

Retrieve named HTTP header.

## Syntax:

```
WEB READ
    HTTPHEADER(data-area)
    NAMELENGTH(data-value)
    VALUE(data-area)
    VALUELENGTH(data-area)
```

## Service:

Web

## Options:

### **HTTPHEADER(data-area)**

Specify the HTTP header to retrieve.

### **NAMELENGTH(data-value)**

Specify the length of the HTTP header.

### **VALUE(data-area)**

Retrieves the value of the HTTP header.

### **VALUELENGTH(data-area)**

Pass the length of the VALUE buffer, retrieves the length of the VALUE.

## Conditions:

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

LENGERR	
1	The passed VALUELENGTH was less than zero (0).
2	The header value was truncated, buffer too small.

NOTFND	
1	The header could not be found.

# WEB READNEXT COOKIE

---

Retrieve the next HTTP cookie.

## Syntax:

```
WEB READNEXT
    COOKIE(data-area) NAMELENGTH(data-area)
    VALUE(data-area) VALUELENGTH(data-area)
    COMMENT(data-area) COMMENTLENGTH(data-area)
    DOMAIN(data-area) DOMAINLENGTH(data-area)
    MAXAGE(data-area)
    PATH(data-area) PATHLENGTH(data-area)
    SECURE(cvda)
    VERSION(data-area)
```

## Service:

Web

## Options:

### **COOKIE(data-area)**

Retrieves the field name.

### **NAMELENGTH(data-area)**

Pass the length of the FORMFIELD buffer, retrieve the length of the FORMFIELD.

### **VALUE(data-area)**

Retrieves the FORMFIELD's value.

### **VALUELENGTH(data-area)**

Pass the length of the VALUE buffer, retrieve the length of the VALUE.

### **COMMENT(data-area)**

Retrieves the cookie's comment. (Comment is not supported by early versions of JEE.)

### **COMMENTLENGTH(data-area)**

Pass the length of the COMMENT buffer, retrieve the length of the retrieved comment.

### **DOMAIN(data-area)**

Retrieves the cookie's domain.

**DOMAINLENGTH(data-area)**

Pass the length of the DOMAIN buffer, retrieve the length of the retrieved domain.

**MAXAGE(data-area)**

Retrieves the maximum age of the cookie.

**PATH(data-area)**

Retrieves the path of the cookie.

**PATHLENGTH(data-area)**

Pass the length of the PATH buffer, retrieve the length of the retrieved path.

**SECURE(data-area)**

Returns SECURE or NOSECURE, reflecting the status of cookie.

**VERSION(data-area)**

Returns the version of the cookie.

**Conditions:**

ENDFILE	The end of the HTTP headers has been reached.
---------	---

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.
4	There was no prior WEB STARTBROWSE COOKIE

LENGERR	
1	The passed VALUELENGTH was less than zero (0).
4	The form field name was truncated, buffer too small.
5	The value was truncated, buffer too small.

## **WEB READNEXT FORMFIELD**

---

Retrieve the next HTTP form field.

**Syntax:**

```

WEB READNEXT
    FORMFIELD(data-area) NAMELENGTH(data-area)
    VALUE(data-area) VALUELENGTH(data-area)
  
```

**Service:**

Web

## Options:

### **FORMFIELD(data-area)**

Retrieves the field name.

### **NAMELENGTH(data-area)**

Pass the length of the FORMFIELD buffer, retrieve the length of the FORMFIELD.

### **VALUE(data-area)**

Retrieves the FORMFIELD's value.

### **VALUELENGTH(data-area)**

Pass the length of the VALUE buffer, retrieve the length of the VALUE.

## Conditions:

ENDFILE	The end of the HTTP headers has been reached.
---------	---

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.
4	There was no prior WEB STARTBROWSE FORMFIELD.

LENGERR	
1	The passed length was less than zero (0).
4	The form field name was truncated, buffer too small.
5	The value was truncated, buffer too small.

## **WEB READNEXT HTTPHEADER**

---

Retrieve the next HTTP header.

## Syntax:

```
WEB READNEXT
    HTTPHEADER(data-area) NAMELENGTH(data-area)
    VALUE(data-area) VALUELENGTH(data-area)
```

## Service:

Web

## Options:

### **HTTPHEADER(data-area)**

Retrieves the HTTP header name.

### **NAMELENGTH(data-area)**

Pass the length of the HTTPHEADER buffer, retrieve the length of the HTTPHEADER.

### **VALUE(data-area)**

Retrieves the HTTP header's value.

### **VALUELENGTH(data-area)**

Pass the length of the VALUE buffer, retrieve the length of the VALUE.

## Conditions:

ENDFILE	The end of the HTTP headers has been reached.
---------	---

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.
4	There was no prior WEB STARTBROWSE HTTPHEADER.

LENGERR	
1	The passed length was less than zero (0).
4	The header name was truncated, buffer too small.
5	The value was truncated, buffer too small.

## WEB RECEIVE

---

Receive all data from the HTTP request.

### Syntax:

WEB RECEIVE

{ INTO(data-area) [MAXLENGTH(data-value)] | SET(ptr-ref) }  
LENGTH(data-area)

[TYPE(cvda)]

[NOTRUNCATE]

[CLNTCODEPAGE(name)]

[HOSTCODEPAGE(name)]

## Service:

Web

## Options:

### **INTO(data-area)**

Specifies the buffer to contain the data.

### **SET(ptr-ref)**

Specifies a pointer, which will then point to the data.

### **LENGTH(data-area)**

Returns the length of the data received.

### **MAXLENGTH(data-value)**

Specifies the maximum length of data to receive.

### **TYPE(cvda)**

Returns HTTP or NONHTTP.

### **NOTRUNCATE**

If the data exceeds the maximum amount, do not truncate; rather, return it on subsequent RECEIVE commands.

### **CLNTCODEPAGE(name)**

Specifies the name of the client's codepage.

HOSTCODEPAGE must also be specified.

### **HOSTCODEPAGE(name)**

Specifies the name of the host's codepage.

CLNTCODEPAGE must also be specified.

## Conditions:

INVREQ	
1	The command is not a Web application.
3	There is no HTTP request.

LENGERR	
1	The passed length was less than zero (0).
2	The length is positive, but exceeded the maximum.

## WEB RETRIEVE

---

Retrieve a document.

**Syntax:**

WEB RETRIEVE DOCTOKEN(data-area)

**Service:**

Web

**Options:****DOCTOKEN(data-area)**

Specifies 16-byte buffer to contain DOCTOKEN of document previously sent using WEB SEND.

**Conditions:**

INVREQ	
1	The command is a not a Web application.
2	There was no prior WEB SEND command.

## WEB SEND

---

Send documents or text to the HTTP client.

**Syntax:**

```

WEB SEND
    {
    DOCTOKEN(name)
    [ CONTENTTYPE(data-area) CONTENTTYPELENGTH(data-value) ] |
    TEXT(data-value) |
    TEXTLINE(data-value) |
    HTML(data-value) |
    HTMLLINE(data-value)
    }
    [CLNTCODEPAGE(name)]
    [STATUSCODE(data-value)]
    [STATUSTEXT(data-area) LENGTH(data-value)]

```

## Service:

Web

## Options:

### **DOCTOKEN(name)**

Specifies the 16-byte name of the document to send to the web browser client. This comes from the document commands.

### **CONTENTTYPE(data-area)**

Specifies a CONTENTTYPE header for the output document.

### **CONTENTTYPELENGTH(data-value)**

Specifies the length of the CONTENTTYPE data.

### **TEXT(data-value)**

Specifies text to send to the web browser client. HTML special characters are automatically escaped, e.g. < to &lt;.

### **TEXTLINE(data-value)**

Specifies text line to send to the web browser client; a <br> is automatically appended. HTML special characters are automatically escaped, e.g. < to &lt;.

### **HTML(data-value)**

Specifies HTML to send to the web browser client. The content is passed through directly to the client.

### **HTMLLINE(data-value)**

Specifies HTML line to send to the web browser client; a <br> is automatically appended. The content is passed through directly to the client.

### **CLNTCODEPAGE(name)**

Specifies the client codepage.

### **STATUSCODE(data-value)**

Specify the HTTP status code; must be valid HTTP status code.

### **STATUSTEXT(data-area)**

Specify text describing the status code.

### **LENGTH(data-value)**

Specify length of STATUSTEXT.

## Conditions:

INVREQ	
1	The command is a not a Web application.

NOTFND	
1	No such document.
7	Unknown client codepage.

## WEB STARTBROWSE COOKIE

---

### Syntax:

WEB STARTBROWSE COOKIE

### Service:

Web

### Options:

None

### Conditions:

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

## WEB STARTBROWSE FORMFIELD

---

Start browsing form fields.

### Syntax:

WEB STARTBROWSE FORMFIELD

### Service:

Web

### Options:

None

**Conditions:**

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

## WEB STARTBROWSE HTTPHEADER

---

Start browsing HTTP headers.

**Syntax:**

WEB STARTBROWSE HTTPHEADER

**Service:**

Web

**Options:**

None

**Conditions:**

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

## WEB WRITE COOKIE

---

Set an HTTP cookie.

A cookie is stored on the client machine, where it may not be stored in a secure manner. So never store secure information within a cookie; rather, use the cookie as a reference to another secure source. Also, supported cookie sizes may be very small, so do not rely on large quantities of information being stored.

**Syntax:**

WEB WRITE

COOKIE(data-area) NAMELENGTH(data-value)

VALUE(data-area) VALUELENGTH(data-value)

COMMENT(data-area) COMMENTLENGTH(data-area)  
DOMAIN(data-area) DOMAINLENGTH(data-area)  
MAXAGE(data-area)  
PATH(data-area) PATHLENGTH(data-area)  
[SECURE(cvda) | SECURE | NOSECURE]  
VERSION(data-area)

## Service:

Web

## Options:

### **COOKIE(data-area)**

Specifies the HTTP cookie name.

The name is an RFC 2109 name, so it may contain only ASCII alphanumeric characters, not white space, commas, or semicolons, and it may not begin with a dollar sign.

### **NAMELENGTH(data-value)**

Specifies the length of the HTTP cookie name.

### **VALUE(data-area)**

Specifies the value of the HTTP cookie.

With Version 0 cookies, the value should not contain white space, brackets, parentheses, equals signs, commas, double quotes, slashes, question marks, at signs, colons or semicolons; empty values may behave differently in different browsers.

### **VALUELENGTH(data-value)**

Specifies the length of the value.

### **COMMENT(data-area)**

Specifies the cookie's comment. The comment describes the cookie, but it is not used for programmatic purposes.

### **COMMENTLENGTH(data-area)**

Specifies the length of the COMMENT buffer.

### **DOMAIN(data-area)**

Specifies the cookie's domain.

### **DOMAINLENGTH(data-area)**

Specifies the length of the DOMAIN buffer.

### **MAXAGE(data-area)**

Specifies the maximum age of the cookie in seconds. -1 indicates until the browser is shutdown.

**PATH(data-area)**

Specifies the path of the cookie.

**PATHLENGTH(data-area)**

Specifies the length of the PATH buffer, retrieve the length of the retrieved path.

**SECURE(data-area)**

Specifies SECURE or NOSECURE, reflecting whether the cookie may be sent over an insecure channel.

**VERSION(data-area)**

Specifies the version of the cookie. Version 0 is the original Netscape specification, Version 1 complies with RFC 2109. RFC 2109 is still considered experimental, so use it only for experimentation, not for production sites.

**Conditions:**

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

## **WEB WRITE HTTPHEADER**

---

Set an HTTP header.

**Syntax:**

```
WEB WRITE
    HTTPHEADER(data-area) NAMELENGTH(data-value)
    VALUE(data-area) VALUELENGTH(data-value)
    ACCUMULATE
```

**Service:**

Web

**Options:**

**HTTPHEADER(data-area)**

Specifies the HTTP header name.

**NAMELENGTH(data-value)**

Specifies the length of the HTTP header name.

**VALUE(data-area)**

Specifies the value of the HTTP header.

**VALUELENGTH(data-value)**

Specifies the length of the value.

**ACCUMULATE**

Add the header rather than replacing it if already present.

**Conditions:**

INVREQ	
1	The command is a not a Web application.
3	There is no HTTP request.

## WRITE

---

Write a record to a file.

**Syntax:**

```

WRITE
    {FILE(data-value) | DATASET(data-value)}
    [MASSINSERT]
    FROM(data-area)
    RIDFLD(data-area)
    [KEYLENGTH(data-value)]
    [SYSID(data-value) LENGTH(data-area) | LENGTH(data-area)]
    [RBA | RRN]
    [NOSUSPEND]

```

**Service:**

File Control

**Setup:**

**file.<filename>=jdbc:< xml\_file\_descriptor.xml>[;<sql\_connection\_name>]**

The `xml_file_descriptor` is an XML file described in the file setup, containing information relating the database columns to the record positions. It must be placed in the resources folder. Placing a `$XFD FILE="filename.xml"` command before a record in the data division will generate a starting template in the listing folder. The `sql_connection_name` is 'file' by default.

**sql.<sql\_connection\_name>.datasource=<jndi\_datasource\_name>**

The file service uses the SQL connection named 'file' by default, but it may be overridden in the file setup. It must be setup in the same manner as all SQL connections.

## Options:

### **DATASET(data-value)**

Specifies the filename the same as FILE, but FILE is preferred.

### **FILE(data-value)**

Specifies the filename. This filename is referred to in the setup via `file.<filename>`. For example, `FILE('abc')` would refer to setup entry `'file.abc=jdbc:abc.xml'`.

### **FROM(data-area)**

Specifies the data record to be written.

### **KEYLENGTH(data-value)**

Specifies the length of the record ID key field.

### **LENGTH(data-area)**

Specifies the length of the from field.

### **MASSINSERT**

Specifies that this write is part of a mass insert of data, where each WRITE contains MASSINSERT.

### **NOSUSPEND**

The read does not suspend on locks.

### **RBA**

Specifies that the key is by relative byte address.

### **RIDFLD(data-area)**

Specifies the record ID field, the key used to search for the record. This data item is mapped to the column marked with an `ridfld` attribute in the XML file descriptor.

### **RRN**

Specifies that the key is by relative record number.

### **SYSID(data-value)**

Specifies the system ID for the file. This should be done instead by setting the appropriate SQL file connection.

### Conditions:

FILENOTFOUN D	The file itself could not be found.
ILLOGIC	A logical error occurred.
INVREQ	Invalid request.
IOERR	An input/output error occurred, such as an invalid SQL query.
LENGERR	The length of the record read exceeds the given INTO buffer's LENGTH.
NOTAUTH	Not authorized.
NOTOPEN	The file is not open.
RECORDBUSY	The record was locked.

## WRITE JOURNALNAME

---

Create a journal record and write it to a journal destination.

### Syntax:

```
WRITE JOURNALNAME(data-value)
      JTYPEID(data-value)
      FROM(data-area) [FLENGTH(data-value)]
      REQID(data-area)
      PREFIX(data-area) PFXLENG(data-value)
      WAIT
      NOSUSPEND
```

### Service:

Journal

### Setup:

```
journal.<name>=<journal_protocol_reference>
```

### Options:

```
JOURNALNAME
JTYPEID
```

**FROM**  
**FLENGTH**  
**REQID**  
**PREFIX**  
**PFXLENG**  
**WAIT**  
**NOSUSPEND**

**Conditions:**

INVREQ	The command is not valid for the environment.
IOERR	An input/output error has occurred in writing the journal record. This occurs when writing, flushing or syncing a file.
JIDERR	The named journal is not available. This condition occurs when there is no journal setup for the named journal, or the named journal's protocol is invalid.
LENGERR	The journal record size is too large.
NOJBUFSP	The journal has no more space. This may also occur if a wait is interrupted.
NOTAUTH	Not authorized to write to this journal. This may occur when opening a file where not allowed.
NOTOPEN	The journal is not open or cannot be opened.

## **WRITE JOURNALNUM**

---

This command has been deprecated in favor of WRITE JOURNAL.

**Syntax:**

```

JOURNAL
    {JFILEID(data-value) | JOURNALNUM(data-value)}
    JTYPEID(data-value)
    FROM(data-area) LENGTH(data-value)
    PREFIX(data-value) PFXLENG(data-value)
    REQID(data-area)
    WAIT
    STARTIO
    NOSUSPEND

```

## Service:

Journal

## Setup:

journal.DFHJnn=journal\_protocol\_reference

## Options:

### **JFILEID or JOURNALNUM**

Binary halfword indicating journal from 1 through 99. The corresponding journal name is DFHJnn, where nn is the two-digit journal number.

### **JTYPEID**

Two-character code.

### **FROM**

Journal record data.

### **LENGTH**

Length of journal record data.

### **PREFIX**

User prefix data.

### **PFXLENG**

User prefix length.

### **REQID**

The transaction system returns a value in this field unique to the task and journal record.

### **WAIT**

Journal record is written synchronously.

### **STARTIO**

Journal record should be written immediately.

### **NOSUSPEND**

If insufficient journal space, return immediately.

## Conditions:

IOERR	An input/output error has occurred in writing the journal record. This occurs when writing, flushing or syncing a file.
JIDERR	The named journal is not available. This condition

	occurs when there is no journal setup for the named journal, or the named journal's protocol is invalid.
LENGERR	The journal record size is too large.
NOJBUFSP	The journal has no more space. This may also occur if a wait is interrupted.
NOTAUTH	Not authorized to write to this journal. This may occur when opening a file where not allowed.
NOTOPEN	The journal is not open or cannot be opened.

## WRITE JOURNALNAME

---

Create a journal record and write it to a journal destination.

### Syntax:

```
WRITE JOURNALNAME(data-value)
      JTYPEID(data-value)
      FROM(data-area) [FLENGTH(data-value)]
      REQID(data-area)
      PREFIX(data-area) PFXLENG(data-value)
      WAIT
      NOSUSPEND
```

### Service:

Journal

### Setup:

```
journal.name=journal_protocol_reference
```

### Options:

**JOURNALNAME**  
**JTYPEID**  
**FROM**  
**FLENGTH**  
**REQID**  
**PREFIX**  
**PFXLENG**  
**WAIT**

## NOSUSPEND

### Conditions:

INVREQ	The command is not valid for the environment.
IOERR	An input/output error has occurred in writing the journal record. This occurs when writing, flushing or syncing a file.
JIDERR	The named journal is not available. This condition occurs when there is no journal setup for the named journal, or the named journal's protocol is invalid.
LENGERR	The journal record size is too large.
NOJBUFSP	The journal has no more space. This may also occur if a wait is interrupted.
NOTAUTH	Not authorized to write to this journal. This may occur when opening a file where not allowed.
NOTOPEN	The journal is not open or cannot be opened.

## WRITE OPERATOR

---

Write a message to the system operator. The system operator is represented by a journal.

### Syntax:

WRITE OPERATOR

TEXT(data-value) [TEXTLENGTH(data-value)]  
[ROUTECODES(data-value) NUMROUTES(data-value)]  
[EVENTUAL | ACTION(cvda) | CRITICAL | IMMEDIATE | operator-reply]

where operator-reply is:

REPLY(data-area) MAXLENGTH(data-value)  
[REPLYLENGTH(data-area)]  
[TIMEOUT(data-value)]

## Service:

Operator

## Setup:

### **operator.route=journal\_name**

The operator destination is a journal name, which in turn is specified through journal.name=journal\_protocol. The referent of the journal\_protocol\_reference is operator.route, by default, which may be modified to operator\_route in some protocols. So, if operator.2 points to a WebLogic journal, then the WebLogic logging catalog will be 'operator\_2'.

## Options:

### **ACTION(cvda)**

The cvda must be one of the following values, or the name of the cvda may be provided directly.

### **EVENTUAL**

The EVENTUAL message is intended for low-priority operator messages.

The journal entry is written using JTYPEID('!E'), PREFIX("Operator, Eventual Action Required"). If selected by default rather than explicitly, the PREFIX("Operator") is used instead.

### **IMMEDIATE**

The IMMEDIATE message is intended for medium-priority operator messages.

The journal entry is written using JTYPEID('!I'), PREFIX("Operator, Immediate Action Required").

### **CRITICAL**

The CRITICAL message is intended for high-priority operator messages.

The journal entry is written using JTYPEID('!C'), PREFIX("Operator, Critical Action Required").

### **MAXLENGTH(data-value)**

Specifies the maximum length of the reply area.

### **NUMROUTES(data-value)**

Specifies the length of the routing table.

### **REPLY(data-area)**

This data-area is filled with the operator reply. Note that many operator protocols do not support a REPLY and will issue an EXPIRED condition under all circumstances.

The journal entry is written using JTYPEID('!R'), PREFIX("Operator, Reply Required").

**REPLYLENGTH(data-area)**

Returns the actual length of the operator reply.

**ROUTECODES(data-value)**

Contains NUMROUTES bytes, each of which is a binary value 0..255. The default route if unspecified is 2. For compatibility with other implementations, restrict the route to a binary value 1..28.

**TEXT(data-value)**

Contains the text sent to the operator destination.

**TEXTLENGTH(data-value)**

Specifies the length of the TEXT sent to the operator destination.

**Conditions:**

The following conditions are allocated to WRITE OPERATOR, but individual journal protocols may produce additional conditions. See the journal protocol for more information on conditions produced.

EXPIRED	RESP2
7	TIMEOUT has occurred
100	REPLY is not supported for the protocol.

INVREQ	RESP2
1	TEXTLENGTH is invalid.
2	NUMROUTES is invalid.
3	ROUTECODES is invalid.
4	MAXLENGTH is invalid.
5	TIMEOUT is invalid.
6	ACTION is invalid.

LENGERR	RESP2
8	Reply was truncated because it is longer than MAXLENGTH.

## **WRITEQ TD**

---

Write data to a transient data queue.

## Syntax:

```
WRITEQ TD QUEUE(name)
        FROM(data-area)
        [LENGTH(data-value)]
        [SYSID(name)]
```

## Service:

Transient Data Control

## Setup:

```
queue[.sysid_value].name=uri
[sysid.name=sysid_value]
```

## Options:

### **QUEUE(name)**

Specifies the name of the transient data queue. The queue must be defined.

### **FROM(data-area)**

Specifies the data area contents to send to the queue.

### **LENGTH(data-value)**

Specifies the length of the data area contents. The default is the length of data-area.

### **SYSID(name)**

Specifies the system ID. This is used to determine which queue definition is used.

## Conditions:

SYSIDERR	An unknown sysid is being used.
QIDERR	
1	There is no such queue definition.
2	The queue could not be obtained.
INVREQ	The queue producer could not be created, for reasons such as the queue being for input only.
IOERR	The message send to the queue failed.

# WRITEQ TS

---

Write data to a temporary storage data queue.

## Syntax:

```
WRITEQ TS {QUEUE(name) | QNAME(name)}  
          FROM(data-area)  
          [LENGTH(data-value)]  
          [SYSID(name)]  
          [AUXILIARY | MAIN]  
          [NOSUSPEND]  
          [NUMITEMS(data-area) | {ITEM(data-area) [REWRITE]}]
```

## Service:

Temporary Storage Control

## Setup:

```
queue[.sysid_value].name[.main | .auxiliary]=uri  
[sysid.name=sysid_value]
```

## Options:

### **QUEUE(name)**

Specifies the name of the transient data queue. The queue must be defined.

### **QNAME(name)**

See QUEUE.

### **FROM(data-area)**

Specifies the data area contents to send to the queue.

### **LENGTH(data-value)**

Specifies the length of the data area contents. The default is the length of data-area. The length must be between 1 and 32763.

### **SYSID(name)**

Specifies the system ID. This is used to determine which queue definition is used.

### **NUMITEMS(data-area)**

Returns the number of items in the queue.

### **ITEM(data-area)**

If REWRITE is not specified, returns the number of items in the queue. This is for compatibility reasons, NUMITEMS is preferred for this case.

If REWRITE is specified, specifies the item number to rewrite.

### **REWRITE**

Rewrite a particular item number, given by ITEM, within the queue.

### **NOSUSPEND**

Raise NOSPACE rather than suspending if there is no remaining room in the queue.

### **MAIN**

Requests that the queue use main storage. It does so by selecting the queue reference ending in .main, if available.

### **AUXILIARY**

Requests that the queue use auxiliary (disk) storage. It does so by selecting the queue reference ending in .auxiliary, if available.

## **Conditions**

SYSIDERR	An unknown sysid is being used.
QIDERR	
1	There is no such queue definition.
2	The queue could not be obtained.
LENGERR	The length is invalid, it must be between 1 and 32763.
NOSPACE	Insufficient space is available to write, and NOSUSPEND was not coded.

## **XCTL**

---

Transfer program control. Link to another program, after ending the current program.

### **Syntax:**

```
XCTL PROGRAM(name)
      [COMMAREA(data-area) [LENGTH(data-area)]]
      [INPUTMSG(data-area) [INPUTMSGLEN(data-value)]]
```

## Service:

Program Control

## Setup:

**pct.name=program\_uri**

program\_name must be a valid program reference, either a classname or URI.

**xlt.name=program\_name**

program\_name must be a valid name in the program table.

## Options:

**PROGRAM(name)**

The program name to be loaded.

**COMMAREA(data-area)**

The COMMAREA to pass to the linked program.

**LENGTH(data-area)**

The length of the COMMAREA.

**INPUTMSG(data-area)**

Specifies data to be received by the other program's RECEIVE.

**INPUTMSGLEN(data-value)**

Specifies the length of the INPUTMSG.

## Conditions:

LENGERR	
11	COMMAREA length not between 0 and 32767.
27	INPUTMSGLEN not between 0 and 32767

PGMIDERR	
1	The program name has no table entry.
3	The program could not be loaded.

# Appendix DFHVALUE(name)

---

Access to special constants is provided in source code by coding DFHVALUE(name), where name is one of the following.

## DFHVALUE constants

---

Num	Name
1	IGNORE
1	NOTAPPLIC
2	BDAM
3	VSAM
3	SFS
4	REMOTE
5	ESDS
6	KSDS
7	RRDS
8	KEYED
9	NOTKEYED
10	BASE
11	PATH
12	FIXED
13	VARIABLE
14	UNDEFINED
15	NOTSUPPORTED
16	BLOCKED
17	UNBLOCKED
18	OPEN
19	CLOSED
20	OPENING
21	CLOSING
22	CLOSEREQUEST
23	ENABLED
24	DISABLED
25	DISABLING
26	OLD
27	SHARE
28	NEW
29	RECOVERABLE
30	NOTRECOVABLE
30	NOTRECOVERABLE
31	EMPTYREQ

<b>Num</b>	<b>Name</b>
32	NOEMPTYREQ
33	UNENABLED
34	UNENABLING
35	READABLE
36	NOTREADABLE
37	UPDATABLE
38	NOTUPDATABLE
39	BROWSABLE
40	NOTBROWSABLE
41	ADDABLE
42	NOTADDABLE
43	DELETABLE
44	NOTDELETABLE
45	HEX
46	DEC
47	BLK
48	EXCTL
49	NOEXCTL
52	ALTERABLE
53	NOTALTERABLE
54	LOG
55	NOLOG
56	CTRLABLE
57	NOTCTRLABLE
60	VTAM
61	BSAM
62	BTAM
62	BTAM-ES
63	BGAM
64	TCAM
65	TCAMSNA
66	CONSOLE
67	CREATE
68	NOCREATE
69	ACQUIRED
70	RELEASED
71	ACQUIRING
72	COLDACQ
73	INSERVICE
74	OUTSERVICE
75	ATI
76	NOATI
77	TTI
78	NOTTI

<b>Num</b>	<b>Name</b>
79	PAGEABLE
80	AUTOPAGEABLE
81	ALLOCATED
82	CONFFREE
83	CONFRECEIVE
84	CONFSEND
85	FREE
86	PENDFREE
87	PENDRECEIVE
88	RECEIVE
89	ROLLBACK
90	SEND
91	SYNCFREE
92	SYNCRECEIVE
93	SYNCSEND
94	FREEING
95	AVAILABLE
96	OBTAINING
100	NOTTABLE
101	CICSTABLE
102	USERTABLE
103	REMTABLE
110	PRIMARY
111	TAKEOVER
121	IRC
122	INDIRECT
123	XM
124	APPC
125	LU61
126	PENDING
127	NOTPENDING
143	XOK
144	XNOTDONE
149	C
150	ASSEMBLER
151	COBOL
152	PLI
152	PL1
154	PROGRAM
155	MAP
155	MAPSET
156	PARTITIONSET
158	ANY
159	BELOW

<b>Num</b>	<b>Name</b>
160	PURGEABLE
161	NOTPURGEABLE
162	OLDCOPY
163	HOLD
164	NOHOLD
165	LPA
166	NOTLPA
167	NEWCOPY
168	PHASEIN
169	ALLCONN
170	AUTOCONN
171	NONAUTOCONN
172	GOINGOUT
173	SHARED
174	PRIVATE
175	SPRSTRACE
175	SPRSPACE
176	STANTRACE
177	SPECTRACE
178	DYNAMIC
179	STATIC
180	STARTUP
181	ACTIVE
182	FIRSTQUIESCE
183	FINALQUIESCE
184	SYSDUMP
185	NOSYSDUMP
186	TRANDUMP
187	NOTRANDUMP
188	SWITCH
192	BACKOUT
194	EXTSECURITY
194	CICSSECURITY
194	SECURE
195	TRANSACTIONSECURITY
196	NOSECURITY
196	NOSECURE
197	ALTERNATE
198	DEFAULT
200	OFF
201	ON
202	RESSECNO
203	RESSECINT
204	RESSECYES

<b>Num</b>	<b>Name</b>
204	RESSECEXT
205	CMDSECNO
207	CMDSECYES
207	CMDSECEXT
208	COMMIT
209	READBACK
210	EMPTY
211	NOTEMPTY
212	FULL
213	TERMINAL
214	NOTERMINAL
215	PHYSICAL
216	LOGICAL
221	EXTRA
222	INTRA
223	NOCTL
224	ASACTL
225	SWITCHING
226	INPUT
227	OUTPUT
228	DISPATCHABLE
229	RUNNING
231	SUSPENDED
233	TASK
234	TERM
235	DEST
236	PURGE
237	FORCEPURGE
241	MCHCTL
242	PRESETSEC
243	NOPRESETSEC
244	SIGNEDON
245	SIGNEDOFF
246	UOW
247	AUXILIARY
248	MAIN
250	TAPE1
251	TAPE2
252	DISK1
253	DISK2
254	DISK2PAUSE
255	SMF
256	OPENINPUT
257	OPENOUTPUT

<b>Num</b>	<b>Name</b>
258	READY
259	NOTREADY
260	CURRENT
261	CLOSELEAVE
262	AUTOARCH
263	NOAUTOARCH
264	REVERTED
265	ADVANCE
266	COLDSTART
267	WARMSTART
268	EMERGENCY
269	LOGTERM
270	FINPUT
271	FOUTPUT
272	FOPEN
273	FCLOSE
274	OK
275	NOWRITE
275	READONLY
276	REMOVE
277	RECOVERED
285	NOSWITCH
286	SWITCHNEXT
287	SWITCHALL
288	SHUTDOWN
289	NOSHUTDOWN
290	RESET
291	ADD
292	DELETE
310	INTSTART
311	INTSTOP
312	AUXSTART
313	AUXPAUSE
314	AUXSTOP
317	GTFSTART
318	GTFSTOP
319	SYSTEMON
320	SYSTEMOFF
321	USERON
322	USEROFF
323	SINGLEON
324	SINGLEOFF
330	PERF
331	NOPERF

<b>Num</b>	<b>Name</b>
332	EXCEPT
333	NOEXCEPT
334	EVENT
335	NOEVENT
340	WAIT
341	NOWAIT
342	FORCE
349	CLOSEFAILED
350	IMMCLOSE
351	FORCECLOSE
352	IMMCLOSING
353	FORCECLOSING
354	FWDRECOVABLE
355	UNDETERMINED
356	NORMALBKOUT
357	FAILEDDBKOUT
358	FAILINGBKOUT
359	INVALID
360	VALID
361	NOTFWDRCVBLE
362	EXITTRACE
363	NOEXITTRACE
364	ZCPTRACE
365	NOZCPTRACE
366	TCEXITALL
367	TCEXITSYSTEM
368	TCEXITNONE
369	TCEXITALLOFF
370	CEDF
370	MODEL
371	NOCEDF
371	SURROGATE
372	SESSION
373	APPC SINGLE
374	APPC PARALLEL
375	COBOLII
376	NOTINIT
377	LE370
378	INACTIVE
379	CICSDATAKEY
380	USERDATAKEY
381	CICSEXECKEY
382	USEREXECKEY
383	DPLSUBSET

<b>Num</b>	<b>Name</b>
384	FULLAPI
385	FMHPARM
386	NOFMHPARM
387	OBOPERID
388	NOOBOPERID
391	APLKYBD
392	NOAPLKYBD
393	APLTEXT
394	NOAPLTEXT
395	AUDALARM
396	NOAUDALARM
397	BACKTRANS
398	NOBACKTRANS
399	COLOR
400	NOCOLOR
401	COPY
402	NOCOPY
403	DUALCASE
404	NODUALCASE
405	EXTENDEDDES
406	NOEXTENDEDDES
407	FORMFEED
408	NOFORMFEED
409	HFORM
410	NOHFORM
411	VFORM
412	NOVFORM
413	HILIGHT
414	NOHILIGHT
415	KATAKANA
416	NOKATAKANA
417	LIGHTPEN
418	NOLIGHTPEN
419	MSRCONTROL
420	NOMSRCONTROL
421	OBFORMAT
422	NOOBFORMAT
423	OUTLINE
424	NOOUTLINE
425	PARTITIONS
426	NOPARTITIONS
427	PRINTADAPT
428	NOPRINTADAPT
429	PROGSYMBOL

<b>Num</b>	<b>Name</b>
430	NOPROGSYMBOL
431	ALLQUERY
432	NOQUERY
433	COLDQUERY
434	SOSI
435	NOSOSI
436	TEXTKYBD
437	NOTEXTKYBD
438	TEXTPRINT
439	NOTEXTPRINT
440	VALIDATION
441	NOVALIDATION
442	RELREQ
443	NORELREQ
444	DISCREQ
445	NODISCREQ
446	ALTPRTCOPY
447	NOALTPRTCOPY
448	PRTCOPY
449	NOPRTCOPY
450	UCTRAN
451	NOUCTRAN
452	TRANIDONLY
485	NEWSESSION
486	OLDSESSION
487	NOSTSN
488	STSNSET
489	STSNTEST
490	EB
491	CD
492	MORE
493	LIC
494	RU
495	NOTCDEB
496	NONE
497	DEFRESP1
498	DEFRESP2
499	DEFRESP3
500	NOALARM
501	ALARM
502	FMH
503	NOFMH
504	PROTECTED
505	UNPROTECTED

<b>Num</b>	<b>Name</b>
506	MDT
507	NOMDT
508	DATA
509	STSN
510	BEGINSESSION
511	TIMEOUT
512	INSTALLFAIL
513	DISCARDFAIL
514	SETFAIL
515	ACQFAIL
516	SESSIONLOST
517	SESSIONFAIL
518	CONVIDLE
519	ADDFAIL
520	DELETEFAIL
521	UNSOLDATA
522	NORMALRESP
523	EXCEPTRESP
524	ATTENTION
525	LUSTAT
526	CANCEL
527	RTR
528	DEFRESP1OR2
529	POSITIVE
530	NEGATIVE
531	NOMSGJRNL
532	INOUT
533	T3278M2
534	T3278M3
535	T3278M4
536	T3278M5
537	T3279M2
538	T3279M3
539	T3279M4
540	T3279M5
541	LUP
542	FORMATTED
543	DATASTREAM
544	LOSE
545	WIN
546	NOTINBOUND
547	INBOUND
549	RELEASING
550	INSTALLED

<b>Num</b>	<b>Name</b>
551	NOTINSTALLED
552	TPS55M2
553	TPS55M3
554	TPS55M4
555	TPS55M5
556	NOCONV
557	PENDSTSN
558	PENDBEGIN
559	APPLICATION
560	PENDDATA
561	PENDSTART
562	PENDRELEASE
563	RELEASE
564	PENDUNSOL
565	PENDPASS
600	CONVERSE
601	NOCONVERSE
602	SYNCPOINT
603	NOSYNCPOINT
604	GMT
605	LOCAL
606	FORMATEDF
607	NOFORMATEDF
608	NOTASKSTART
609	STARTED
610	STOPPED
611	TASKSTART
612	BUSY
613	NOTBUSY
614	SCS
615	DS3270
616	ASCII7
617	ASCII8
618	AUTOSTART
620	INDOUBT
621	INFLIGHT
622	WAITFORGET
623	CONTROLSHUT
623	DB2
623	SQL
624	CANCELLED
625	FIRSTINIT
626	SECONDINIT
627	THIRDINIT

<b>Num</b>	<b>Name</b>
628	INITCOMPLETE
629	STANDBY
630	AUTOACTIVE
631	AUTOINACTIVE
632	CTLGALL
633	CTLGMODIFY
634	CTLGNONE
635	START
636	POST
637	DELAY
638	ROUTE
640	CLEAR
641	NOCLEAR
642	USER
643	SYSTEM
644	SHUTENABLED
645	SHUTDISABLED
650	EXCI
651	GENERIC
652	SPECIFIC
653	SYSCONNECT
654	NOSYSCONNECT
655	FORCECANCEL
657	NOISOLATE
658	ISOLATE
659	NOTDEFINED
660	CICS
661	NONCICS
663	SUBSPACE
664	BASESPACE
665	XCF
666	REQUIRED
667	NOTREQUIRED
668	SOS
669	NOTSOS
670	REGISTERED
671	UNREGISTERED
672	UNAVAILABLE
673	CMDPROT
674	NOCMDPROT
675	RELATED
677	REGERROR
678	DEREGISTERED
679	DEREGERROR

<b>Num</b>	<b>Name</b>
680	REENTPROT
681	NOREENTPROT
682	SOSBELOW
683	SOSABOVE
684	DAE
685	NODAE
690	CONNECTED
691	NOTCONNECTED
693	SPI
694	NOSPI
700	NORECOVDATA
701	RECOVDATA
702	RESYNC
703	UNCONNECTED
706	IMMQUIESCED
707	QUIESCED
708	QUIESCING
709	UNQUIESCED
710	NOLOSTLOCKS
711	NORETAINED
712	RECOVERLOCKS
713	REMLOSTLOCKS
714	RESETLOCKS
715	RETAINED
716	RETRY
720	RLS
721	NOTRLS
722	UNCOMMITTED
723	CONSISTENT
724	REPEATABLE
730	RLSACTIVE
731	RLSINACTIVE
732	VRRDS
740	REMSESSION
751	EXECENQ
752	EXECENQADDR
753	OWNER
754	WAITER
755	CONNECTION
756	DATASET
757	HEURCOMMIT
758	HEURBACKOUT
761	RLSSERVER
762	SHUNTED

<b>Num</b>	<b>Name</b>
763	WAITCOMMIT
765	WAITING
766	WAITRMI
767	TDQ
768	TSQ
770	COORDINATOR
771	RMI
772	STARTING
773	SUBORDINATE
774	NRS
780	MVS
781	DUMMY
782	FAILED
783	FLUSH
784	NOSYSLOG
785	SYSLOG
788	COLD
789	INITIAL
790	ENDAFFINITY
791	CACHE
792	COMMITFAIL
793	DATASETFULL
794	DEADLOCK
795	DELEXITERROR
796	INDEXRECFULL
797	IOERROR
798	OPENERERROR
799	RLSGONE
800	BACKUPNONBWO
810	SKIP
811	REWIND
811	LEAVE
812	REREAD
813	MOD
814	QUEUE
815	REJECT
816	IGNORERR
820	UNATTEMPTED
830	RRCOMMITFAIL
831	RRINDOUBT
832	LCKSTRUCFULL
833	CFTABLE
834	LOAD
835	NOLOAD

<b>Num</b>	<b>Name</b>
836	CONTENTION
837	LOCKING
900	ABEND
902	CGROUP
903	CONNECT
904	CONNECTING
905	CSIGN
906	CTERM
907	CTX
908	COPID
909	CUSERID
910	DISCONNING
911	EQUAL
912	EXIT
913	GROUP
914	HIGH
915	LOW
916	NOCONNECT
917	NOEXIT
918	NORELEASE
919	NOROLLBACK
920	NOTWAIT
921	N906D
922	POOL
923	REBUILD
924	RECONNECT
925	SIGN
926	SQLCODE
927	TWAIT
928	TXID
929	TX
930	USERID
931	N906
932	TPOOL
933	OPID
1000	NOTFIRED
1001	FIRED
1002	ACTIVITY
1003	COMPOSITE
1004	TIMER
1005	AND
1006	OR
1010	PROCESS
1011	NOUSER

<b>Num</b>	<b>Name</b>
1012	EVENTFAIL
1013	FORCED
1014	INCOMPLETE
1015	UNEXPECTED
1016	NORMAL
1017	EXPIRED
1018	UNEXPIRED
1021	NOTDYNAMIC
1022	NOTROUTABLE
1023	ROUTABLE
1024	DORMANT
1025	CANCELLING
1026	COMPLETE
1027	NOTSUSPENDED
1030	SSL
1031	NOSSL
1032	CLIENTAUTH
1033	HTTPYES
1034	HTTPNO
1036	APPEND
1037	NOAPPEND
1038	BINARY
1039	EBCDIC
1050	QUASIRENT
1051	THREADSAFE
1052	BASEAPI
1053	OPENAPI
1054	NOFORCE
1055	CKOPEN
1056	UKOPEN
1057	QR
1058	INTERNAL
1071	NOAUTO
1072	PROGAUTO
1073	FULLAUTO
1080	JVM
1081	NOJVM
1082	DEBUG
1083	NODEBUG
1084	NONLE370
2000	HTTP
2001	NONHTTP

## DFHVALUE(name) Device Commands

---

Num	Name
1	T7770
2	SYSTEM7
8	TCONSOLE
18	SEQDISK
20	MAGTAPE
24	CDRDLPRT
32	HARDCOPY
33	TWX3335
34	TELETYPE
36	T1050
40	T2740
42	T2741COR
43	T2741BCD
64	VIDEOTERM
65	T2260L
72	T2260R
74	T1053
76	T2265
80	TTCAM
128	BISYNCH
130	T2770
132	T2780
133	T3780
134	T2980
136	T3735
137	T3740
138	T3600BI
145	T3277R
146	T3275R
147	T3284R
148	T3286R
153	T3277L
155	T3284L
156	T3286L
160	BIPROG
161	SYSTEM3
164	SYS370
166	SYS7BSCA
176	SDLC
177	T3601

<b>Num</b>	<b>Name</b>
178	T3614
180	T3790
181	T3790UP
182	T3790SCSP
184	T3650PIPE
185	T3653HOST
186	T3650ATT
187	T3650USER
189	CONTNLU
190	INTACTLU
191	BATCHLU
192	LUTYPE6
193	LUTYPE4
208	RESSYS
209	ISCMMCONV
210	LUCMODGRP
211	LUCSESS

# Appendix DFHRESP

---

RESP codes are available symbolically using DFHRESP(name), such as DFHRESP(NORMAL).

<b>Num</b>	<b>Name</b>
0	NORMAL
1	ERROR
2	RDATT
3	WRBRK
4	EOF
5	EODS
6	EOC
7	INBFMH
8	ENDINPT
9	NONVAL
10	NOSTART
11	TERMIDERR
12	FILENOTFOUND
13	NOTFND
14	DUPREC
15	DUPKEY
16	INVREQ
17	IOERR
18	NOSPACE
19	NOTOPEN
20	ENDFILE
21	ILLOGIC
22	LENGERR
23	QZERO
24	SIGNAL
25	QBUSY
26	ITEMERR
27	PGMIDERR
28	TRANSIDERR
29	ENDDATA
30	INVTREQ
31	EXPIRED
32	RETPAGE
33	RTEFAIL
34	RTESOME
35	TSIOERR

<b>Num</b>	<b>Name</b>
36	MAPFAIL
37	INVERRTERM
38	INVMPSZ
39	IGREQID
40	OVERFLOW
41	INVLDC
42	NOSTG
43	JIDERR
44	QIDERR
45	NOJBUFSP
46	DSSTAT
47	SELNERR
48	FUNCERR
49	UNEXPIN
50	NOPASSBKRD
51	NOPASSBKWR
52	SEGIDERR
53	SYSIDERR
54	ISCINVREQ
55	ENQBUSY
56	ENVDEFERR
57	IGREQCD
58	SESSIONERR
59	SYSBUSY
60	SESSBUSY
61	NOTALLOC
62	CBIDERR
63	INVEXITREQ
64	INVPARTNSET
65	INVPARTN
66	PARTNFAIL
69	USERIDERR
70	NOTAUTH
71	VOLIDERR
72	SUPPRESSED
80	NOSPOOL
81	TERMERR
82	ROLLEDBACK
83	END
84	DISABLED
85	ALLOCERR
86	STRELERR
87	OPENERR
88	SPOLBUSY

<b>Num</b>	<b>Name</b>
89	SPOLEERR
90	NODEIDERR
91	TASKIDERR
92	TCIDERR
93	DSNNOTFOUND
94	LOADING
95	MODELIDERR
96	OUTDESCRERR
97	PARTNERIDERR
98	PROFILEIDERR
99	NETNAMERR
100	LOCKED
101	RECORDBUSY
102	UOWNNOTFOUND
102	UOWNNOTFOUND
103	UOWLNOTFOUND
104	LINKABEND
105	CHANGED
106	PROCESSBUSY
107	ACTIVITYBUSY
108	PROCESSERR
109	ACTIVITYERR
110	CONTAINERERR
111	EVENTERR
112	TOKENERR
113	NOTFINISHED
114	POOLERR
115	TIMERERR
116	SYMBOLERR
117	TEMPLATERR

# Appendix EJB Setup Files

---

To setup remote EJB usage of the transaction, several .xml files may be required, including ejb-jar.xml and possibly one or more required for a particular application server. Samples of these are included.

These files are automatically generated by the IDE during deployment.

## ejb-jar.xml

```
<ejb-jar>
<enterprise-beans>
<session>
  <ejb-name>LinkDispatcher</ejb-name>
  <home>com.heirloomcomputing.etp.transaction.env.ejb.LinkDispatcherHome</home>
  <remote>com.heirloomcomputing.etp.transaction.env.ejb.LinkDispatcher</remote>
  <local-home>
    com.heirloomcomputing.etp.transaction.env.ejb.LinkDispatcherLocalHome
  </local-home>
  <local>com.heirloomcomputing.etp.transaction.env.ejb.LinkDispatcherLocal</local>
  <ejb-class>com.heirloomcomputing.etp.transaction.env.ejb.LinkDispatcherEJB</ejb-class>

  <session-type>Stateful</session-type>
  <transaction-type>Bean</transaction-type>

  <env-entry>
    <env-entry-name>xlt.ejb_transid</env-entry-name>
    <env-entry-type>java.lang.String</env-entry-type>
    <env-entry-value>bound_program_name</env-entry-value>
  </env-entry>
  <env-entry>
    <env-entry-name>pct.bound_program_name</env-entry-name>
    <env-entry-type>java.lang.String</env-entry-type>
    <env-entry-value>program_id</env-entry-value>
  </env-entry>
</session>
</enterprise-beans>
</ejb-jar>
```

## weblogic-ejb-jar.xml

```
<?xml version="1.0"?>
<!DOCTYPE weblogic-ejb-jar PUBLIC
"-//BEA Systems, Inc.//DTD WebLogic 6.0.0 EJB//EN"
"http://www.bea.com/servers/wls600/dtd/weblogic-ejb-jar.dtd">

<weblogic-ejb-jar>

  <weblogic-enterprise-bean>
    <ejb-name>LinkDispatcher</ejb-name>
    <jndi-name>remote_jndi_name</jndi-name>
  </weblogic-enterprise-bean>

</weblogic-ejb-jar>
```

# Appendix Unsupported Commands

---

The Elastic Transaction Platform supports the CICS API command set with the following exceptions:

## Distributed Processing Service:

---

RECEIVE	ISSUE ERASEAUP	EXTRACT PROCESS
SEND	ISSUE ENDFILE	ISSUE ABEND
CONVERSE	ISSUE PRINT	CONNECT PROCESS
ISSUE EODS	ISSUE SIGNAL	ISSUE
ISSUE COPY	ALLOCATE	CONFIRMATION
WAIT TERMINAL	FREE	ISSUE ERROR
ISSUE LOAD	POINT	ISSUE PREPARE
WAIT SIGNAL	BUILD ATTACH	ISSUE PASS
ISSUE RESET	EXTRACT ATTACH	EXTRACT LOGONMSG
ISSUE DISCONNECT	EXTRACT TCT	EXTRACT
ISSUE ENDOUTPUT	WAIT CONVID	ATTRIBUTES

## Interval Control:

---

START BREXIT

## Batch Data Interchange:

---

ISSUE ADD	ISSUE QUERY	ISSUE WAIT
ISSUE ERASE	ISSUE END	ISSUE SEND
ISSUE REPLACE	ISSUE RECEIVE	
ISSUE ABORT	ISSUE NOTE	

## TCP/IP:

---

EXTRACT CERTIFICATE

## Trace Control:

---

MONITOR

## **Task Management Service:**

---

WAITCICS

WAIT EXTERNAL