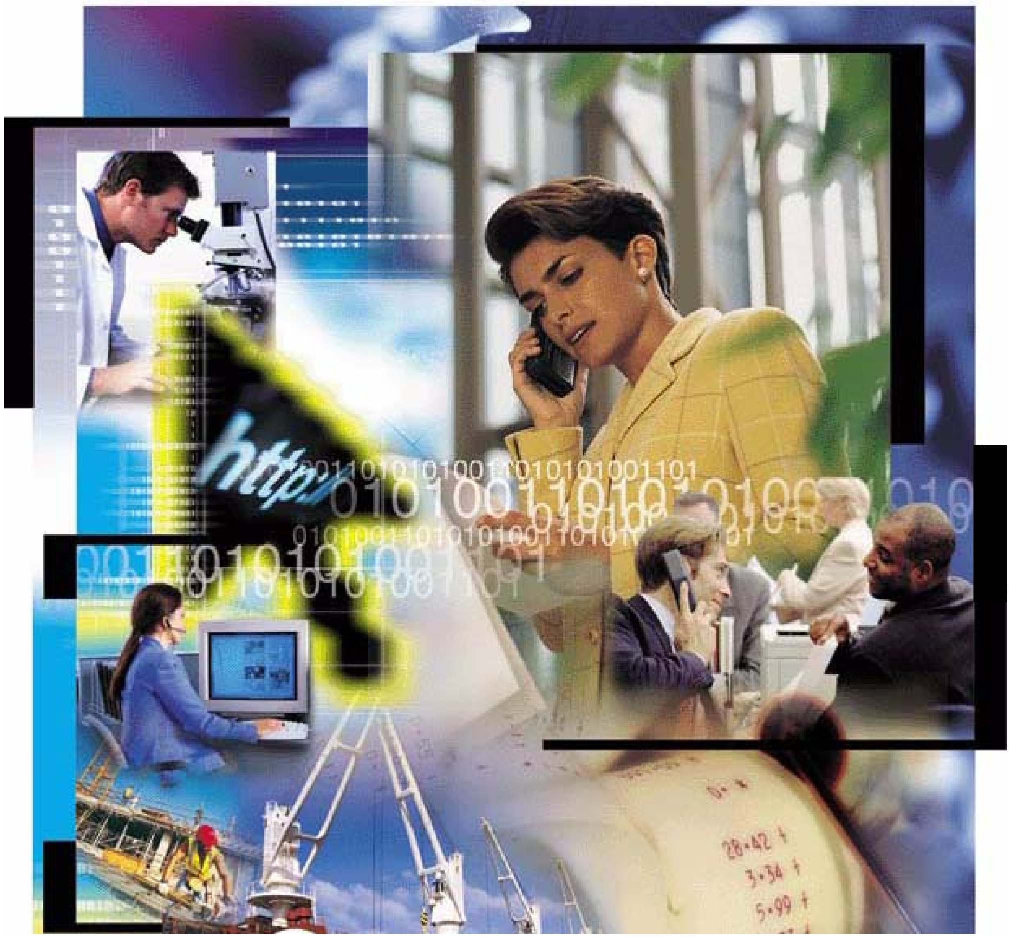
GXS EDI Services

Expedite Base/MVS Programming Guide

Version 4 Release 6



GC34-2204-05

Sixth Edition (November 2005)

This edition applies to Expedite Base/MVS, Version 4 Release 6, and replaces GC34-2204-04.

Contents

To the reader

This book gives you the information necessary to program and use Expedite Base/MVS for your company's applications. Expedite Base/MVS is an Information Exchange base communication program that enables you to transmit data files and messages to and from Information Exchange. Users of Expedite Base/MVS can use the features of Information Exchange through the AT&T Global Network or the Internet.

Who should read this book

This book is written for experienced MVS programmers familiar with Information Exchange who want to write application programs for Expedite Base/MVS.

Terminology conventions

The term *network* as used in this book refers to the network provided by AT&T Global Network Services in the United States and other countries.

In this book, *baud rate* and *data rate* are synonymous.

The term *compression* as used in this book refers to data compression and decompression performed with the TDAccess product provided by Click Commerce, Inc., which may not be available in all countries.

Type conventions

Understanding the type conventions used in this book can help you apply the material covered.

All application program interface (API) commands and parameters are displayed in small, uppercase letters; for example, AUTOMODE. In the examples, commands and parameters display in lowercase.

In the step-by-step instructions in this book, boldfaced type is used to instruct you to:

- Type in specific information. For example, "Type **a:\setup.exe**."
- Click objects on the screen or press specific keys. For example, "Click **Next**."

Blank lines have been added to some examples to help you use them. So, some examples in the book may not look exactly like what you see when you use Expedite Base.

In Chapter 9, "Expedite Base/MVS response records," the command format examples have the following type conventions:

- Required parameters and values display in boldface type.
- Default values are underlined.
- Parameter values are italicized.

NOTE: When *blank* is listed as a variable, it refers to a blank space and not the actual typed word.

In general, you do not have to worry about case when typing commands and parameters, and can use uppercase or lowercase letters. However, there are exceptions, which are documented where they occur.

There are many syntax examples in the book. These examples display in a different typeface from the rest of the text. Required parameters and values are shown in boldface type in the syntax examples. Parameter variables are printed in italics, and default values are underlined.

fileid(*fileid*) format(<u>*n*</u>|*y*)

Because you do not have to worry about case when typing API commands, all examples are illustrated for you in lowercase letters as shown below. You can use uppercase or lowercase letters as you wish when typing API commands, although you must follow JCL requirements when using EXEC commands. For specifics on EXEC commands, see "EXEC statement" on page 21.

```
audit account(123at) userid(name);
```

Words that are in the glossary are shown in italics the first time they are used in the body of the book.

To locate information in this book, use the table of contents, the list that is shown below under the heading "How this book is organized," and the index.

How this book is organized

This book contains eleven chapters, eight appendixes, a glossary, and an index:

- Chapter 1, "Introducing Expedite Base/MVS," introduces Information Exchange and Expedite Base/MVS. It describes how Expedite Base/MVS connects to Information Exchange.

- Chapter 2, “Setting up files, including the JCL,” describes the four basic files upon which the Expedite Base/MVS system is based and tells you how to create them. It also describes some optional work files that you might need to create, depending on which Expedite Base/MVS features you want to use.
- Chapter 3, “Communicating with other operating systems,” provides information on transferring files to other systems, including ASCII interfaces, host interfaces, and older Information Exchange interfaces.
- Chapter 4, “Sending and receiving EDI data,” tells you how to send and receive EDI-formatted data. It includes a description of the EDI destination resolution procedure.
- Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery,”describes the Information Exchange checkpoint-level recovery and file-level recovery features. It also explains how you can restart an Information Exchange session.
- Chapter 6, “Using session-level recovery,” describes the Information Exchange session-level recovery feature, which is the default recovery level.
- Chapter 8, “Additional features of Expedite Base/MVS,” describes more Expedite Base/MVS features, free-format messages, and validations for addresses, payment levels, and authorizations.
- Chapter 7, “Expedite Base/MVS commands,” describes the command sequence. It also gives detailed information about the Expedite Base/MVS profile and message file commands.
- Chapter 9, “Expedite Base/MVS response records,” describes the Expedite Base/MVS response records and their formats.
- Chapter 10, “Expedite Base/MVS installation overview,” provides you with general information about how Expedite Base/MVS is installed.
- Chapter 11, “Communicating with Information Exchange using SSL,” describes how to communicate with Information Exchange using SSL with TCP/IP.
- Appendix A, “Product examples,” provides you with three samples of the basic command files.
- Appendix B, “Common data header,” contains information on the common data header (CDH) used by Expedite Base/MVS to communicate information about files to other interfaces.
- Appendix C, “Sample audit report,” tells you how to use the audit command and contains a sample audit report.
- Appendix D, “Expedite Base/MVS messages and codes,” contains all Expedite Base/MVS return codes with messages, explanations, and advice on the actions that you should take in response to these codes.
- Appendix E, “Using data compression,” describes the use of compression software to compress and decompress data with Expedite products.
- Appendix F, “Migrating from IBM b,” compares and contrasts Expedite Base/MVS to expEDIte/MVS Host Release 3.
- Appendix G, “Batch data interface,” describes using Batch Data Interface (BDI) from a remote job entry (RJE) terminal.

Sample files

Sample files are provided with the product software. For file names, see your systems programmer.

Summary of changes

The following are the major changes to recent releases of Expedite Base/MVS.

Application Hosting - EDI Services Expedite software for MVS is enhanced to do client-authenticated Secure Sockets Layer (SSL) over TCP/IP using the AT&T Global Network or the Internet to connect to the Information Exchange mailbox component of GXS EDI Services, formerly IBM EDI Services. This enhancement is not supported in all locations.

EDI Services customers who want the value-added features provided by an Expedite communication client can now take advantage of enhanced authentication and data privacy when connecting to Information Exchange using this new SSL TCP/IP connectivity.

Highlights of Expedite Base/MVS Version 4 Release 6 client-authenticated SSL communications with Information Exchange include the following:

- Use of an Internet Public Key Infrastructure (PKI) Client X.509 certificate
- Certificate Management using either gskkyman under the UNIX® System Services or RACF
- Use of System SSL which is part of the System SSL Cryptographic Services Base element of z/OS
- Use of new IP addresses when communicating with Information Exchange using an SSL connection
- Changes to the Expedite Base/MVS Job Control Language (JCL)
- Command enhancements:
  - New SSL command enables users to specify SSL TCP/IP connectivity
  - New KEYRINGFILE/KEYRINGSTASHFILE/KEYRINGPASSWORD parameters on the IDENTIFY command to communicate information concerning the X.509 certificate
  - New KEYRINGFILE/KEYRINGSTASHFILE/KEYRINGPASSWORD parameters on the START command to communicate information concerning the X.509 certificate

Related books

The following books contain information related to the topics covered in this guide:

- *Information Exchange Interface Programming Guide, GC34-2222*
- *Information Exchange Charges Reference, GX66-0653*
- *Information Exchange Messages and Formats, GC34-2324*
- *Information Exchange Administration Services User's Guide, GC34-2221*
- *System SSL Programming, SC24-5901*

- *Expedite Base Commands Reference, GC34-2328*
- *Comm-Press User's Guide*

For your convenience, many of these documents can be viewed on the EDI Services Web site library page at: http://www.gxsolc.com/edi_bes.html.

Chapter 1

Introducing Expedite Base/MVS

Information Exchange is an electronic mailbox system that you can use to send information to and receive information from trading partners. Expedite Base/MVS provides the interface that makes it possible to use Information Exchange from a z/OS environment.

Expedite Base/MVS uses Information Exchange to deliver and receive data, such as electronic data interchange (EDI) data. To communicate with Expedite Base/MVS and transfer data files in and out of Information Exchange, you use application program interface (API) commands. You place these commands in control files.

Expedite Base/MVS supports several Information Exchange data recovery methods. You can direct Expedite Base/MVS to recover data at the level of a session, a checkpoint, or a file, or you can manage the data recovery yourself.

Information Exchange

Information Exchange provides a means of sending, storing, and retrieving information electronically and makes it possible for users on dissimilar computer systems to communicate with each other. By establishing a computer-to-computer communication network between different locations, Information Exchange can both speed and simplify the delivery of messages, EDI envelopes, and other data.

Information Exchange is an electronic mailbox system that is part of EDI Services. With Information Exchange as an alternative to terminal-to-computer communication, you send messages to the Information Exchange mailbox and retrieve the messages waiting for you in the mailbox.

Information Exchange can link the geographically scattered locations of a single company or of different companies; for example, a manufacturing company can use Information Exchange to communicate with its suppliers or distributors.

A program running on your computer can communicate with Information Exchange on behalf of one or more Information Exchange addresses. This means that your computer can communicate with any Information Exchange address, through Information Exchange, during one physical connection between systems.

Expedite users of Information Exchange can also take advantage of the strong authentication and encryption offered by an SSL connection over TCP/IP. Expedite users must obtain an X.509 certificate for each Information Exchange address that will be used in session with Information Exchange using Expedite and must store the certificates on their computer.

NOTE: SSL communication is not available in all locations. Contact your local representative for further information.

Understanding the Expedite Base/MVS operating environment

Expedite Base/MVS runs on a z/OS operating system, which uses extended binary-coded decimal interchange code (EBCDIC). Expedite Base/MVS can communicate with some American National Standard Code for Information Interchange (ASCII) systems, such as a personal computer (PC), as well as with other EBCDIC computers.

Hardware requirements

- An IBM eServer zSeries server capable of running z/OS V1.4 is required.
- Additional hardware may be required to connect to Information Exchange. Contact your Information Exchange network provider.
- DASD requirements will vary depending on the Expedite Base/MVS features used. For example, if checkpoint-level recovery is chosen, additional files are required to successfully complete a session.

Software requirements

- Expedite Base/MVS can run on z/OS V1.4 with APAR PQ85252.
- The following items are used by Expedite Base/MVS but are considered base elements of z/OS V1.4:
  - **Communications Server:** Provides both TCP/IP and SNA support. Expedite Base/MVS uses one of these protocols to communicate with Information Exchange.
  - **Cryptographic Services:** Expedite Base/MVS uses facilities of this base element to support SSL communications. Component System Secure Sockets Layer is used for secure communications as well as certificate management.
  - **Language Environment (LE):** Expedite Base/MVS requires the runtime environment provided by LE.
- For SSL TCP/IP communications, a certificate is required. For information about obtaining a certificate, see Chapter 11, “Communicating with Information Exchange using SSL.”

Connecting to Information Exchange

To connect to Information Exchange, you must have an Information Exchange account, user ID, and password. These items must be defined in the Expedite Base/MVS profile. Use the IDENTIFY command to provide Expedite Base/MVS with these values.

To connect to Information Exchange using SSL over TCP/IP communications, you also need an X.509 certificate stored on your computer for each Information Exchange account and user ID that you will be using during your Expedite sessions. You can obtain an X.509 certificate at the following Web site: https://pki.services.ibm.com/. The certificate is provided on either the IDENTIFY or START commands.

For more information, see Chapter 7, "Expedite Base/MVS commands."

Starting an Information Exchange session

Before you can transfer data, you must start an Information Exchange session. Usually, Expedite Base/MVS starts this session automatically with information provided in the IDENTIFY command. However, you can start a session manually by using the START command.

Before using the START command, specify AUTOSTART(n) in the TRANSMIT command to disable the automatic session start function. When the session starts, the connection to Information Exchange is complete. At this point, Expedite Base/MVS performs data transfer requests.

ATTENTION: If you start an Information Exchange session while another Information Exchange session with the same account and user ID is running, the results are unpredictable. You should not use the same Information Exchange account and user ID for more than one Expedite interface.

For more information on the START and TRANSMIT commands, see Chapter 7, "Expedite Base/MVS commands."

Ending an Information Exchange session

When all file transfer requests are complete, you must end the Information Exchange session. Usually, Expedite Base/MVS ends the session automatically. However, if you specified AUTOEND(n) in the TRANSMIT command to disable the automatic session end function before starting the session, you must end the session manually with the END command.

For more information on the END command, see "END command" on page 89.

Giving commands to Expedite Base/MVS

You use command files to provide Expedite Base/MVS with input. The commands you enter in the profile command (INPRO) and message command (INMSG) files give Expedite Base/MVS the information it needs to operate. INPRO is the data definition name (ddname) of the profile command file that tells Expedite Base/MVS how to establish a communication environment. INMSG is the ddname of the message command file. Use INMSG to send and receive files and messages, including EDI data.

For detailed information on commands, see Chapter 7, "Expedite Base/MVS commands."

Getting responses from Expedite Base/MVS

Expedite Base/MVS places response records in two response files. It generates the profile response file (OUTPRO) when it processes an INPRO file and generates the message response file (OUTMSG) when it processes an INMSG file.

OUTPRO repeats INPRO commands and associates appropriate return codes to these commands. You can examine the RETURN record in OUTPRO to see if a particular profile command parsed correctly.

OUTMSG works similarly to OUTPRO. Expedite Base/MVS places response records in OUTMSG when it processes the INMSG file. OUTMSG contains the completed commands and the associated return codes. Check the RETURN record in OUTMSG to be sure that your files and messages have been sent correctly.

For detailed information on response records, see Chapter 9, “Expedite Base/MVS response records.”

Understanding command syntax

You place application program interface (API) commands in the profile command and message command files (INPRO and INMSG) to communicate with Expedite Base/MVS. Expedite Base/MVS places response records in the profile response and message response files (OUTPRO and OUTMSG) to tell you if your API commands have completed. You can use uppercase or lowercase letters for API commands. An example of Expedite Base/MVS command syntax is shown below.

```
#Comment or description command parameter(value) parameter(value) ...
   parameter(value);
```

Descriptions of the syntax elements are given below:

#   Defines or delimits comments. Type any information after #. The program ignores the characters after # until it encounters a new line. You do not have to end comments with #. Expedite Base/MVS considers the end of the line in which the comment exists to be the end of that comment. If you want a comment to continue, begin a new line with #.

If you place a command after the comment on the same line, Expedite Base/MVS ignores that command. To ensure that your command is processed, you should either place the command on a new line or place the command before the comment.

Some file names and description statements contain #. If you include a # in a parameter value, Expedite Base/MVS knows from the context that the # is part of a command and does not ignore the parameter value or the characters that follow.

The following example shows how to place a command and a comment on the same line. In this case, Expedite Base/MVS ignores all the information after #. The program begins processing again at the ACCOUNT parameter because this parameter is on a new line that does not begin with #.

```
send fileid(data.file) #this is a new file account(acct2)
userid(user2) class(class1);
```

command   Identifies the Expedite Base/MVS command.

parameter   Identifies a parameter associated with the command preceding it.

value   Defines the value associated with the parameter.

...   Indicates in the example above that you can specify as many parameters as necessary. This is not part of the syntax.

;	Ends the command.

You can type Expedite Base/MVS commands and parameters in uppercase or lowercase letters. The commands and parameters can span several lines in a command file. However, the following limitations apply:

- You must type the entire command name (for example, IDENTIFY) on a single input line.
- You must type the entire parameter name (for example, IEACCOUNT) on a single input line.
- A left parenthesis must immediately follow each parameter. Do not use spaces between parameter names and values. For example, type **ieaccount**(acct) rather than **ieaccount** (acct).
- You must end each command with a semicolon.

Identifying Expedite Base/MVS error messages

Error messages generated by Expedite Base/MVS are written to OUTPRO and OUTMSG. They reflect errors resulting from faulty commands or system communication problems. Each error message is identified by a code.

For more information on error messages, see Appendix D, "Expedite Base/MVS messages and codes."

Identifying Information Exchange error messages

Information Exchange may generate error messages. The most common reason for an error message is that Information Exchange was unable to deliver a file. Information Exchange puts these messages into your mailbox; they are given a sender's account ID of *SYSTEM* and user ID of *ERRMSG*. You can identify these error messages by their codes. You must receive these messages with a RECEIVE or RECEIVESTREAM command.

Identifying Expedite Base/MVS completion codes

Expedite Base/MVS (the IEBASE program) ends with the following completion codes: 00, 04, 08, and 12. For details on the meanings of these codes, see Appendix D, "Expedite Base/MVS messages and codes."

Sending and receiving data

You can use three sets of commands to send and receive messages, files, and EDI data using Expedite Base/MVS:

- SEND RECEIVE
- SENDEDI RECEIVEEDI
- SENDSTREAM RECEIVESTREAM

With the SEND command, you can send each message or file individually. You can send and receive binary files, which contain machine instructions such as executable computer programs for a computer to read or text files, which contain information for people to read.

The RECEIVE command retrieves either all files or specific files, including free-format messages, from the Information Exchange mailbox.

The SENDEDI command transmits multiple EDI envelopes with different addresses from a single file. Information in this file can consist of any combination of X12, Uniform Communication Standard (UCS), Electronic Data Interchange For Administration, Commerce, and Transport (EDIFACT), or United Nations/Trade Data Interchange (UN/TDI) data. SENDEDI resolves the destinations of the different pieces of information from the EDI data, so you do not have to retype existing destination information.

The RECEIVEEDI command can receive multiple EDI envelopes containing different types of data.

The SENDSTREAM command sends data from the message command file (INMSG) to Information Exchange. The RECEIVESTREAM command receives data directly into the message response file. SENDSTREAM and RECEIVESTREAM are valid only when you use session-level recovery.

For more information on sending and receiving data, see Chapter 3, "Communicating with other operating systems," and Chapter 4, "Sending and receiving EDI data."

Send and Receive file number limits

Information Exchange limits the number of files sent and received between commits. The size of the files does not matter. The current limit of 1,000 is an Information Exchange value that can be set differently in different Information Exchange installations. Contact Customer Care to determine the maximum for Information Exchange installations outside the U.S.

If you use session-level recovery, you cannot send or receive more than 1,000 files in a session. If you are sending EDI data, each EDI envelope counts as a file, even if you have multiple EDI envelopes in a single file.

If you use user-level recovery, you must not specify more than 1,000 SEND, SENDEDI, or PUTMEMBER commands without specifying a COMMIT command. Do not send more than 1,000 EDI envelopes from a single file, because each envelope counts as a file. You should not have a problem with the receive file limit with user-level recovery.

If you use checkpoint-level recovery, you should not have any problems, unless you send more than 1,000 small files where the total number of characters sent is less than your COMMITDATA value on the TRANSMIT command in your profile. The default value is 141,000. If this is a problem, lower the COMMITDATA value. You should not have a problem with the receive file limit with checkpoint-level recovery.

If you use file-level recovery, you will not have problems with the limit on the sends or receives. Each file is committed as it is sent or received, and the maximum number of files between commits is 1. If you are sending or receiving many small files, you can get better performance using checkpoint-level recovery.

If you try to send more files than the limit, you get an error from Information Exchange, which Expedite reports to you as error 31360 or 29999 on the SESSIONEND record. In this case, break your input file into multiple input files and run Expedite repeatedly until the files are sent. For session-level recovery, see Chapter 6, "Using session-level recovery." For all other recovery levels, see Chapter 5, "Using checkpoint-level, file-level, and user-initiated recovery."

If you have more than 1,000 files in your mailbox that match your receive requests and you are using session-level recovery, Information Exchange stops sending files when the 1,000 limit is reached. If you have more files in your mailbox, Expedite returns a 28171 value in the RETURN parameter after the last file is received. The SESSIONEND is 28010, indicating the session completed successfully but not all commands were processed. The data you already received is no longer in your mailbox. You do not need to recover the session that ended with 28171 and 28010.

If your session ends with 28010 and the last receive command ended with RETURN(28171), you should process the data received or make sure it is not overwritten if you run Expedite again. For more information on session-level recovery, see Chapter 6, “Using session-level recovery.”

If you switch to checkpoint-, user-, or file-level recovery, you can receive all the files in your mailbox in a single session without getting the 28171 return code.

Free-format messages

You can create free-format messages using the editor on your MVS system. Use the SEND and RECEIVE commands to transfer these messages. For more information, see “Free-format messages” on page 139.

Acknowledgments

Although they are not error messages, Information Exchange also identifies Information Exchange acknowledgments with a sender’s account name of *SYSTEM* and a user ID of *ERRMSG*.

The three types of acknowledgments you can request on any of the Expedite Base/MVS SEND commands are described below:

Acknowledgment type	Description
receipt	Generated by Information Exchange when a message reaches the receiver’s mailbox after a successful Expedite Base/MVS session.
delivery	Generated by Information Exchange when a destination user receives a message from the Information Exchange mailbox.
purge	Generated by Information Exchange when a message is purged from the receiver’s mailbox.

Restart and recovery considerations

Expedite Base/MVS supports four types of data recovery: session-level, checkpoint-level, file-level, and user-initiated.

When you use session-level recovery and an error occurs, you must retransmit all data. Although retransmitting all data is time consuming, it is the easiest way to recover from errors. For this reason, session-level recovery is the default recovery level of Expedite Base/MVS. For more information on session-level recovery, see Chapter 6, “Using session-level recovery.”

Checkpoint-level recovery, file-level recovery, and user-initiated recovery require more effort on your part (for example, setting up special work files) but can result in more efficient data recovery. For more information on these recovery levels, see Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery.”

Providing security

The network provides security at the network access level, the application selection level, and the data access level. You should understand that network security features operate within a widely used data processing environment. Information Exchange can protect users only if they observe security controls. Passwords must be changed at intervals to ensure mailbox security. Authorization levels enable users to perform certain functions and restrict those users from other functions.

If additional network security is desired, you can configure Expedite Base/MVS to connect with Information Exchange using an SSL connection over TCP/IP. To use an SSL connection, you must obtain and store an X.509 certificate on your computer. You must also have the necessary TCP/IP connectivity to Information Exchange. For more information, see Chapter 11, “Communicating with Information Exchange using SSL.”

Selecting the Extended Security Option

The Extended Security Option (ESO) provides additional password and Information Exchange mailbox security. The ESO USER flag may be turned on by the Information Exchange Service Administrator, using Information Exchange Administration Services. Users with ESO can send files and messages to users with ESO and users without ESO.

For information on using ESO, refer to the *Information Exchange Administration Services User's Guide*.

ESO contains the following security features:

- Your Information Exchange Service Administrator must change your ESO password if it is the same as your user ID. If you do not provide a new password in the START command, your Information Exchange session will not be successful.
- Your new ESO password must conform to the following rules:
  - Must not contain the user ID as any part
  - Must be at least six characters in length
  - Must contain at least three different characters
  - Must contain a nonnumeric first and last character
  - Must contain at least one non-alphabetic character
  - Must contain at least one alphabetic character
  - Must contain only the valid characters A-Z, 0-9, and special characters # @ and $
  - Must not contain more than two identical consecutive characters
  - Must be different from the current or five previous passwords
  - Must not contain more than three identical consecutive characters from the previous password

If your new password does not conform to these rules, it is considered invalid, and your Information Exchange session will not be successful.

- Information Exchange revokes your ESO user ID if you make three consecutive attempts to start an Information Exchange session with an invalid password. All further attempts to start a session will be unsuccessful until your Information Exchange Service Administrator resets your password.
- The Information Exchange Service Administrator can use the Information Exchange Administration Services password reset function to reinstate an ESO user ID.

NOTE: Passwords are reset immediately after resetting the ESO option to Y. When the administrator sets the ESO option to Y, all affected users must change their password at their next logon. This is true even if the administrator subsequently resets the ESO option back to N.

Working with libraries

A library is a place to store information for an extended period of time. Unlike messages in a user's mailbox, information in a library is not deleted automatically after a certain amount of time. Libraries are made up of library members, which contain the information you want to store. To use libraries in Expedite Base/MVS:

1. Create the library using Information Exchange Administration Services.
2. Use the PUTMEMBER and GETMEMBER commands to put members into the library and retrieve them from the library, placing them into your mailbox.
3. Use the LISTLIBRARIES and LISTMEMBERS commands to identify libraries and members to which you have access.

For more information on libraries, see “Working with libraries” on page 141, and refer to the library information in the *Information Exchange Administration Services User’s Guide*.

Chapter 2

Setting up files, including the JCL

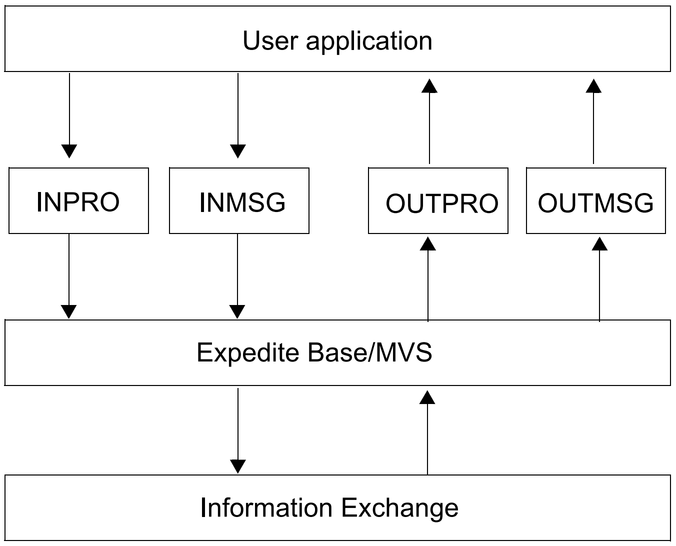
This chapter gives you the information you need to set up Expedite Base/MVS before you begin to use it. It provides a general explanation of how Expedite Base/MVS works, describes the files you need to create and their attributes, or properties, and contains sample JCL statements.

How Expedite Base/MVS uses its primary files

Expedite Base/MVS is a file-oriented program. To use Expedite Base/MVS, you place requests in a command file, run the IEBASE program, and examine the appropriate response file to find completed requests. The primary command and response files used by Expedite Base/MVS are listed here:

- Profile command file (ddname INPRO)
- Profile response file (ddname OUTPRO)
- Message command file (ddname INMSG)
- Message response file (ddname OUTMSG)

The following figure illustrates how these files make it possible for you to communicate with Information Exchange.



Before you can use Expedite Base/MVS, you must create these primary files and any other work files and optional files needed for your applications.

Expedite Base/MVS files

You must create all files used by Expedite Base/MVS before you can use the program. Expedite Base/MVS does not create any files for you. The four primary command and response files, INPRO, OUTPRO, INMSG, and OUTMSG, are required. You need to create other files too, depending on whether you decide to use checkpoint or session-level recovery, which traces your request, and the amount of error description you want.

File limitations

You must consider the following limitations, which apply to all files used with Expedite Base/MVS. These limitations apply in addition to any other limitations given for specific files.

- Use only sequential files and partitioned data set (PDS) members. Do not use virtual storage access method (VSAM) files.
- Do not specify DISP=MOD with PDS members.
- Use DISP=MOD with care when not otherwise restricted. Partially received data is not erased when you run the job again or restart it. If you use DISP=MOD, you must remove the data before running the same job again to avoid duplication.
- Do not use spanned records in PDS members.
- Do not use RECFM=U.
- Do not make response files, such as OUTMSG, members of the same PDS as the file received.

- When creating the input to a file, do not use a “numbers on” text-editing option. Create the file without line numbers.
- Files referenced on the SEND or SENDEDI commands (in the FILEID parameter) must not be empty.

Descriptions of required files

The files all users are required to create before using Expedite Base/MVS are described here. You might need to create other files as well, depending on how your application will interface with Expedite Notification Manager.

Profile command file (INPRO)

The ddname of the profile command file is INPRO. You must create an INPRO file before you can use Expedite Base/MVS. Enter setup information in INPRO using profile commands. Expedite Base/MVS uses the information contained in INPRO to connect to Information Exchange and set up the communication environment. You can find more details on profile commands in “Profile commands” on page 66.

Expedite Base/MVS uses default values for many of the profile command parameters; however, it does not use default values for the required parameters. If you plan to use AUTOSTART(y), the default option, and ENABLESSL(y) on the TRANSMIT command, Expedite Base/MVS requires several parameters from the IDENTIFY command. You must also provide X.509 certificate information using the KEYRINGFILE or KEYRINGFILE, KEYRINGPWD, and KEYRINGSTASHFILE parameters. Which parameters are required depends on how you manage your certificates. These parameters are optional if either AUTOSTART(n) or ENABLESSL(n) are used. For more information, see Chapter 11, “Communicating with Information Exchange using SSL.”

If you plan to use COMMTYPE(s), the default option of the TRANSMIT command, the SNACOMM profile command is also required.

If you plan to use COMMTYPE(t), the TCPCOMM profile command is required. The SNACOMM and TCPCOMM commands each have at least one required parameter.

Required IDENTIFY parameters

If Expedite Base/MVS starts the Information Exchange session automatically, which is the default, you must use the IDENTIFY command to provide the following parameters so Expedite Base/MVS can identify you as a user. If you use AUTOSTART(n) in the TRANSMIT command, these parameters are optional.

IEACCOUNT	The Information Exchange account.
IEUSERID	The Information Exchange user ID.
IEPASSWORD	The Information Exchange password For more information, see “IDENTIFY command” on page 68.

In addition, if you use AUTOSTART(y) and ENABLESSL(y), you must provide X.509 certificate information using the following parameters: KEYRINGFILE or KEYRINGFILE plus KEYRINGPASSWORD or KEYRINGSTASHFILE. The parameters that are required depend on how you manage your X.509 certificate. For more information, see Chapter 11, “Communicating with Information Exchange using SSL.”

Required SNACOMM parameter

If you plan to use SNA communications, which is the default on the TRANSMIT command, use the USERLUNAME parameter of the SNACOMM command to indicate the names of the Expedite Base/MVS logical units (LUs) you are using. Your system programmer can provide you with these LU names. For more information, see “SNACOMM command” on page 71.

Required TCPCOMM parameters

If you plan to use TCP/IP communications by specifying COMMTYPE(t) on the TRANSMIT command, you must provide the following parameters to indicate the address of the Information Exchange system you will be using. The sample file EXXIPADD lists the valid values by geographic location.

IETCPHOSTn	The Information Exchange IP address
IETCPPORTn	The Information Exchange port

NOTE: For SSL communications, COMMTYPE(t) is required.

For more information, see “TCPCOMM command” on page 73.

INPRO attributes

Create INPRO with the following attributes:

This attribute:	Has these characteristics:
Space	INPRO has no minimum space requirement. It must be large enough to hold all of your profile commands.
Record format	INPRO can have fixed or variable records. Variable records are usually more efficient.
Record length	Record length must not exceed 255 bytes.

ATTENTION: When creating the input to this file, do not use a “numbers on” text-editing option. Create this file without line numbers.

Sample INPRO

A sample INPRO file is shown here. This file contains the minimum IDENTIFY and SNACOMM command requirements when you use AUTOSTART(y).

```
identify ieaccount(acct) ieuserid(user01) iepassword(xxxxxx);
snacomm userluname(luname);
```

Profile response file (OUTPRO)

The ddname of the profile response file is OUTPRO. You must create an OUTPRO file before you can use Expedite Base/MVS. When Expedite Base/MVS reads the profile command file, it echoes the profile commands to OUTPRO, along with their associated return codes. The return record in OUTPRO contains the return codes for each command. The PROFILERC record contains the return code for the processing of the entire INPRO file.

NOTE: When the information is to be sent to the job output, the ddname of the profile response file should point to SYSOUT=* rather than to a data file.

For more information on response records, see Chapter 9, “Expedite Base/MVS response records.”

OUTPRO attributes

Create OUTPRO with the following attributes:

This attribute:	**Has these characteristics:**
Space	OUTPRO must be large enough to hold the echoed INPRO commands and their associated responses.
Record format	OUTPRO can have fixed or variable records. Variable records are usually more efficient.
Record length	Record length must be at least as long a that in INPRO, at least 72 bytes long, but less than or equal to 255 bytes.

Sample OUTPRO

The following OUTPRO file is produced by the command file shown in “Sample INPRO” on page 14.

```
identify ieaccount(acct) ieuserid(user01) iepassword(xxxxx);
RETURN(00000);
snacomm userluname(luname);
RETURN(00000);
PROFILERC(00000);
```

Message command file (INMSG)

The ddname name of the message command file is INMSG. You must create an INMSG file before you can use Expedite Base/MVS. You can send and receive files by entering file transfer requests with message commands into INMSG. For more information on message commands, see “Message commands” on page 77.

INMSG attributes

Create INMSG with the following attributes:

This attribute:	Has these characteristics:
Space	INMSG has no minimum space requirement. It must be large enough to hold all of your message commands.
Record format	INMSG can have fixed or variable records. Variable records are usually more efficient.
Record length	Record length must not exceed 255 bytes.

ATTENTION: When creating the input to this file, do not use a “numbers on” text-editing option. Create this file without line numbers.

Sample INMSG

A sample of an INMSG used to send an electronic data interchange (EDI) file called DEPT01.EDIDATA.FILE is provided here. This example uses an Information Exchange user class of EDITEST. A user class identifies documents to your trading partners, the business associates with whom you exchange electronic information. After sending this file, you receive all the files with a user class of EDITEST from the Information Exchange mailbox.

```
sendedi fileid(dept01.edidata.file) class(editest);
receiveedi fileid(dept01.ediout.file) class(editest);
```

Message response file (OUTMSG)

The ddname of the message response file is OUTMSG. You must create an OUTMSG file before you can use Expedite Base/MVS. When Expedite Base/MVS processes the message command file, it echoes the completed commands to OUTMSG along with their associated return codes. The RETURN record in OUTMSG contains the return codes for each completed command.

If a RETURN record with a zero return code follows a command, the command completed successfully. If a RETURN record with a nonzero return code follows a command, the command did not complete successfully. The SESSIONEND record contains the final return code for INMSG and INPRO files that have been processed.

NOTE: When the information is to be sent to the job output, the ddname of the message response file should point to SYSOUT=* rather than to a data file.

For more information on processing OUTMSG, see Chapter 6, “Using session-level recovery,” and Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery.”

OUTMSG attributes

Create OUTMSG with the following attributes:

This attribute:	Has these characteristics:
Space	OUTMSG must be large enough to hold the echoed INMSG commands and their associated responses.
Record format	OUTMSG can have fixed or variable records. Variable records are usually more efficient.
Record length	Record length must be at least as long as that in INMSG, at least 72 bytes long, and less than or equal to 255 bytes.
Data recovery limitations	If you are using checkpoint-level, file-level, or user-initiated recovery, OUTMSG must be a sequential data set on a direct access storage device (DASD), not a PDS member or tape data set. DISP=MOD is not permitted.

Sample OUTMSG

The following OUTMSG file is produced after the successful completion of the INMSG file shown in “Message command file (INMSG)” on page 15 is:

```
AUTOSTART SESSIONKEY(3JKKDIR8);
STARTED LASTSESS(0) RESPCODE(00000) SESSIONKEY(3JKKDIR8)
IEVERSION(04) IERELEASE(07);
RETURN(00000);
```

```
sendedi fileid(dept01.edidata.file) class(editest);
SENT UNIQUEID(57586838) LENGTH(2260) ACCOUNT(ATAP) USERID(DEPT01)
EDITYPE(X12)
DESTINATION(DUNS01) QUALIFIER(01) CONTROLNUM(000022223) CLASS(EDITEST)
MSGNAME(00022223) MSGSEQNO(00001);
RETURN(00000);
receiveedi fileid(dept01.ediout.file) class(editest);
RECEIVED ACCOUNT(ATAP) USERID(DEPT01) CLASS(EDITEST) CHARGE(5)
RECEIVER(ACT1 DEPT01) RECVQUAL(01)
SENDER(ACT2 DEPT02) SENDQUAL(01) CONTROLNUM(000022228)
FILEID(DEPT01.EDIOUT.FILE) MSGDATE(980325) MSGDATELONG(19980325)
MSGTIME(192507)
MSGSEQO(0000009) MSGNAME(00022223) MSGSEQNO(00001) SESSIONKEY(3JKKDIR8)
DELIMITED(E) TIMEZONE(L) SYSNAME(EXMV000T) SYSLEVEL(450) DATATYPE(E)
EDITYPE(X12) SENDERFILE(DEPT01.EDIDATA.FILE) SENDERLOC(3390 /TSO203)
FILEDATE(980325) FILEDATELONG(19980325) RECFM(F) RECLEN(80)
RECDLM(E) UNIQUEID(57586838) SYSTYPE(21) SYSVER(4);
RETURN(00000);
AUTOEND;
RETURN(00000);
SESSIONEND(00000);
```

Received files

You must create files to hold the data that Expedite Base/MVS receives from Information Exchange. Be sure to make these files large enough to hold all of the data that they need to receive. Also, make sure that the logical record length (LRECL) and record format (RECFM) of the files is appropriate for the received data.

Data recovery files

You must create the following files if you plan to use checkpoint-level, file-level, or user-initiated recovery:

- Output work file (ddname OUTWORK)
- Receive work file (ddname RCVWORK)
- Session file (ddname SESSION)
- EDI work file (ddname EDIWORK). This file is only needed when using the SENDEDI command with the VERIFY parameter equal to c or g.

For more information on data recovery files, see Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery.”

Optional files

You might want to use some of the optional Expedite Base/MVS files. These files are described here.

Error message file (ERRORMSG)

Expedite Base/MVS can provide error descriptions in the RETURN, WARNING, SESSIONEND, and PROFILERC response records. These error descriptions are contained in the ERRORMSG file.

If you refer to this file with the ddname ERRORMSG when you run Expedite Base/MVS, Expedite Base/MVS writes error descriptions to response records when errors occur. If you do not include this file when you run Expedite Base/MVS, error message codes display without error message text. We suggest you make use of the ERRORMSG file. Your system programmer can provide you with the file name to use.

CAUTION: Do not modify this file. Modifications can cause you to receive incorrect error messages.

Extended error text file (ERRORTXT)

Expedite Base/MVS can provide extended error descriptions in the RETURN, WARNING, and SESSIONEND response records. These extended error descriptions are contained in the ERRORTXT file.

If you refer to this file with the ddname ERRORTXT when you run Expedite Base/MVS, Expedite Base/MVS includes error messages in the response records when errors occur. If you do not include this file when you run Expedite Base/MVS, error messages display without extended error text. We suggest you make use of the ERRORTXT file. Your system programmer can provide you with the file name to use.

CAUTION: Do not modify this file. Modifications can cause you to receive incorrect error messages.

Base trace file (BASETRC)

You can request several types of traces, separately or in combination. All traces, except the LINK trace, go to the same file.

The base trace file, BASETRC, contains trace information other than link trace. If you have requested a trace using the TRACE profile command, refer to this file with the ddname BASETRC when you run Expedite Base/MVS. You need to request a trace only when you are working with Customer Care on a problem. For more information on how to select traces, see “TRACE command” on page 74.

BASETRC attributes

Create BASETRC with the following attributes:

This attribute:	Has these characteristics:
Space	The space BASETRC requires varies, depending on the trace options you request and the amount of data you transmit. If you request a PROTOCOL trace, you must make BASETRC large enough to hold all of the data you plan to send or receive during an Expedite Base/MVS run, in addition to other trace information that Expedite Base/MVS generates.
Record format	BASETRC can have fixed or variable records. Variable records are usually more efficient.
Record length	BASETRC record length must be at least 72 bytes. A record length of 255 bytes is usually effective.
Data set organization	BASETRC must be a sequential data set. It cannot be a member of a PDS.

Link trace file (LINKTRC)

This file contains information produced by the link trace. If you request a link trace using the LINK parameter of the TRACE profile command, create this file and give it ddname LINKTRC. You need to request a link trace only if you are working with Customer Care on a problem.

LINKTRC attributes

Create LINKTRC with the following attributes:

This attribute:	Has these characteristics:
Space	The space LINKTRC requires varies, depending on the amount of data you transmit. LINKTRC must be large enough to hold all of the data you plan to send or receive during an Expedite Base/MVS run, in addition to other trace information Expedite Base/MVS generates.
Record format	LINKTRC record format should be set to FBA.
Record length	LINKTRC record length must be between 121 and 255 bytes. If the record length is too short, Expedite Base/MVS writes a message to the job log and does not write a link trace. A record length of 255 bytes is usually effective.
Data set organization	LINKTRC must be a sequential data set. It cannot be a member of a PDS.

Trace message text (TRACEMSG)

Expedite Base/MVS can provide trace message text in languages other than English. Expedite Base/MVS uses this file to change trace file text for national language support.

If you refer to this file with the ddname TRACEMSG when you run Expedite Base/MVS, it takes trace messages from this file. If you receive a version of Expedite Base/MVS customized for a language other than English, your system programmer can provide you with this file name.

NOTE: This file is only included in versions of Expedite Base/MVS that are not in the English language.

Electronic data interchange (EDI) control files

The SENDEDI command uses the following files when sending EDI data:

```
EDI qualifier table (QUALTBL)
EDI destination tables (TTABLExx)
```

For more information about these files, see Chapter 4, "Sending and receiving EDI data."

Job control language (JCL) sample

A sample of the JCL used to start Expedite Base/MVS is provided here. This example uses checkpoint-level recovery. All parameters that you must enter exactly are provided in the following example using uppercase letters. JCL variables specific to your system or a job are shown in lowercase letters.

```
//jobname JOB (insert your installation-specific job statement)
//JOBLIB DD DSN=expedite.mvs.loadlib,DISP=SHR
//expstep EXEC PGM=IEBASE,REGION=0K
//SYSPRINT DD sysout=*
//SYSOUT DD sysout=*
//*
//*command files
//INMSG DD DSN=user1.message.command,DISP=SHR
//INPRO DD DSN=user1.profile.command,DISP=SHR
//*
//*response files
//OUTMSG DD DSN=user1.message.response,DISP=OLD
//OUTPRO DD DSN=user1.profile.response,DISP=OLD
//*
//*data recovery files
//OUTWORK DD DSN=user1.output.work,DISP=OLD
//SESSION DD DSN=user1.session.file,DISP=OLD
//RCVWORK DD DSN=user1.receive.workfile,DISP=OLD
//EDIWORK DD DSN=user1.ediwork.work,DISP=OLD
//*
//*edi control files
//QUALTBL DD DSN=user1.ediqual.table,DISP=SHR
//TTABLExx DD DSN=user1.ttablexx.table,DISP=SHR
//*
//*trace files
//BASETRC DD DSN=user1.iebase.trace,DISP=OLD
//LINKTRC DD DSN=user1.line.trace,DISP=OLD
//*
//*error text files
//ERRORMSG DD DSN=expedite.error.messages(EXXMSG),DISP=SHR
//ERRORTXT DD DSN=expedite.error.messages(EXXTXT),DISP=SHR
//TRACEMSG DD DSN=expedite.error.tracemsg,DISP=SHR
```

EXEC statement

The EXEC statement instructs MVS to start the Expedite Base/MVS program, which is named IEBASE. All EXEC statement parameters must be entered in uppercase characters. Your EXEC statement must include the REGION parameter, and you might also want to use the PARM parameter.

Parameter	Description	
REGION	This parameter may be necessary depending on your z/OS system requirements. It is recommended that you specify 0K for this parameter. This will allow the job to acquire more region space, if necessary, within the constraints of your z/OS system. If necessary, code this on your EXEC or JOB statement.	
PARM	You can code the following keyword parameters:	
	`PARM='POSIX(ON)/ RESET INPRO(INPRO) OUTPRO(OUTPRO) INMSG(INMSG) OUTMSG(OUTMSG) STATUS'`	
	The parameters you can enter are described in the following list:	
	POSIX(ON)/	Establishes the environment necessary to use Expedite Base/MVS with an SSL connection.
	RESET	Resets a checkpoint-level recovery session if you have an unrecoverable error and need to reset your session.
	INPRO	Changes the default ddname for the profile command file, which normally has a ddname of INPRO.
	OUTPRO	Changes the default ddname for the profile response file, which normally has a ddname of OUTPRO.
	INMSG	Changes the default DD name for the message command file, which normally has a ddname of INMSG.
	OUTMSG	Changes the default ddname for the message response file, which normally has a ddname of OUTMSG.
	STATUS	Tells you whether or not Expedite Base/MVS is in a restart situation. No data is sent or received when you issue this parameter. You receive a completion code of 1 if Expedite Base/MVS would try to restart the previous session. You receive a completion code of 0 if Expedite Base/MVS would not try to restart the previous session.

The following example shows an EXEC statement used to change the ddnames of INPRO, OUTPRO, INMSG, and OUTMSG.

```
//EXPSTEP EXEC PGM=IEBASE,REGION=0K,
//PARM='INPRO(INPRO1) OUTPRO(OUTPRO1) INMSG(INMSG1) OUTMSG(OUTMSG1)'
```

Chapter 3

Communicating with other operating systems

When transferring files between different operating systems, you must consider:

- The type of system that will receive the files; for example, MVS, OS/400 or Microsoft Windows
- The record format of the file being sent
- Whether a binary or text file is being sent

This chapter addresses each of the above issues and the actions you should take in each instance. It also discusses a valuable aid to file transfer, the common data header (CDH).

NOTE: In this chapter, system refers to a particular type of operating system, rather than to a particular Information Exchange system.

How Expedite Base/MVS uses the CDH

Most Information Exchange interfaces can send and receive additional information describing the contents of a file. The CDH contains this information. Typically, it includes the items listed here:

- The type of file (text or binary)
- The record delimiters used in the file, for example, carriage return/line feed (CRLF) characters or record lengths
- Whether the data is EDI-formatted
- The original name of the file on the sending system
- A free-format text description of the file
- Other information describing the original file

When Expedite Base/MVS receives a file, it can use the information in the CDH to format the file. Expedite Base/MVS places the CDH information for received data in the output file on the RECEIVED record or the AVAILABLE record. (See page 165 for a description of the RECEIVED record and page 151 for a description of the AVAILABLE record.) You do not have to do anything to send or receive the CDH because Expedite Base/MVS does this automatically. However, you need to understand the CDH to fully understand the file transfer capabilities of Expedite Base/MVS. For more information on the CDH, see Appendix B, “Common data header.”

Communicating with ASCII operating systems

Expedite Base/MVS runs on an EBCDIC system. Expedite for Windows, Expedite Base for Windows, and Expedite Base/AIX products run on ASCII systems.

This section tells you how to communicate with these ASCII systems.

Sending text files to an ASCII operating system

Text files usually contain data that you can read, rather than executable programs or computerized drawings that are meant for a computer to read. When you send text data to an ASCII system, you must tell Expedite Base/MVS to put carriage-return and line-feed characters (X'0D0A') at the end of each record. Carriage-return and line-feed characters enable the file to maintain its original record format and be readable when it gets to the ASCII system. To do this, use the DELIMIT(C) option of the SEND command; for example,

```
send fileid(user01.mytext.file) account(act1) userid(user04)
class(pctext) delimit(c);
```

The Expedite Base product that receives the data translates that data from EBCDIC to ASCII when it receives the file. You do not have to do anything to request this translation.

Receiving text files from an ASCII operating system

When Expedite Base/MVS receives text files, it looks at the CDH to determine whether or not the data has carriage-return and line-feed characters at the end of each record. If the CDH indicates that these characters are present, Expedite Base/MVS removes them and ends the records where the characters were found.

Because all interfaces do not send a CDH, you must tell Expedite Base/MVS what to do with records received if there is no CDH. To do this, use the DELIMITED(C) parameter of the RECEIVE command. The DELIMITED(C) parameter tells Expedite Base/MVS to assume that the data has carriage-return and line-feed characters when there is no CDH. This permits you to receive text data from older Expedite interfaces that do not send a CDH.

The following is an example of how to use the DELIMITED(C) parameter of the RECEIVE command.

```
receive fileid(user01.textout.file) account(act1) userid(user05)
class(pctext) delimited(c);
```

The sending Expedite Base program on the ASCII system translates the data from ASCII to EBCDIC when it sends the file. You do not have to do anything to request this translation.

Sending binary files to an ASCII operating system

Binary files usually contain data that you cannot read, such as executable programs and computerized drawings. When you send binary files to an ASCII system, you must tell the Expedite Base product receiving the data not to translate the data from EBCDIC to ASCII. To do this, use the DATATYPE(B) option of the SEND command. When you send binary data, do not use the

DELIMIT parameter of the SEND command to tell Expedite Base/MVS to use delimiters to mark records. These characters may be confused with the actual data in the binary file. The following is an example of how to use the DATATYPE(B) option of the SEND command.

```
send fileid(user01.binary.file) account(act1) userid(user04)
   class(pcbin) datatype(B);
```

NOTE: expEDIte/PC (Release 2 or earlier) and Personal Computer Information Exchange Interface (PCIE) cannot correctly receive binary data without an additional data translation program.

Receiving binary files from an ASCII operating system

You do not need to use any special RECEIVE parameters to receive binary files. Do not specify DELIMITED(C) on the RECEIVE command when receiving binary data. Specifying DELIMITED(C) tells Expedite Base/MVS to look for carriage-return and line-feed characters at the end of the record. These characters may be confused with the actual data in the binary file. The following is an example of how to use the RECEIVE command when receiving binary data.

```
receive fileid(user01.binary.data) account(act1) userid(user05)
   class(pcbin);
```

The sender on an ASCII system must tell the Expedite Base product that the data is binary. The Expedite Base product then sends the data without translating it from ASCII to EBCDIC.

Communicating with host operating systems

This section tells you how to communicate with other host systems using interfaces such as Expedite/CICS.

Sending text files to a host operating system

Text files usually contain data that you can read. When you send text data to another host system, make certain the file will be readable on the receiving system. There are two ways to do this by using the DELIMIT option of the SEND command:

DELIMIT(L)	This option causes Expedite Base/MVS to insert a 2-byte record length at the beginning of each record. The receiving host system then removes the lengths and ends the records as the lengths indicated.
DELIMIT(C)	This option causes Expedite Base/MVS to insert carriage-return and line-feed characters (X'0D0A') at the end of each record. The receiving host system then removes these characters and ends the records where indicated by the delimiters before their removal.

NOTE: Text files sent with the DELIMIT(C) option are also readable on ASCII systems.

The following is an example of how to use the DELIMIT(C) option of the SEND command.

```
send fileid(user01.mytext.file) account(act1) userid(user03)
   class(text) delimit(c);
```

Receiving text files from a host operating system

When Expedite Base/MVS receives files, it looks at the CDH to determine whether or not the data contains record delimiters, such as carriage-return and line-feed characters. If the CDH indicates that these delimiters are present, Expedite Base/MVS removes them, but ends the records where indicated by the delimiters before their removal.

Because all interfaces do not send a CDH, you must tell Expedite Base/MVS what to do with records received if there is no CDH. To do this, use the DELIMITED parameter of the RECEIVE command. The value that you specify on the DELIMITED parameter must correspond to the way the sending host system delimited the records; for example:

```
receive fileid(user01.textout.file) account(act1) userid(user05)
class(text) delimited(corresponds to sender’s method);
```

Sending binary files to a host operating system

Binary files usually contain data that you cannot read, such as executable programs and computerized drawings. When you send binary data to another host, use the DATATYPE(B) option of the SEND command. If you need to delimit records when you send the data, use the DELIMIT(L) option of the SEND command. Do not use the DELIMIT(C) option of the SEND command when sending binary data. The DELIMIT(C) option tells Expedite Base/MVS to put carriage-return and line-feed characters at the end of every record. These characters may be confused with the actual data in the binary file. The following is an example of how to use the DELIMIT(L) option of the SEND command:

```
send fileid(user01.binary.file) account(act1) userid(user04)
   class(hostbin) datatype(B) delimit(l);
```

Receiving binary files from a host operating system

When receiving files, Expedite Base/MVS looks at the CDH to determine whether or not the data is delimited. If the data is delimited, Expedite Base/MVS uses the delimit type specified in the CDH to format the records when it receives the data.

Because all interfaces do not send a CDH, you must tell Expedite Base/MVS what to do with records received if there is no CDH. To do this, use the DELIMITED parameter of the RECEIVE command. The value you specify on the DELIMITED parameter must correspond with the way the sending host system delimited the records. Do not specify DELIMITED(C) in the RECEIVE command when receiving binary data. Specifying DELIMITED(C) tells Expedite Base/MVS to look for carriage-return and line-feed characters at the end of the record. These characters may be confused with the actual data in the binary file. The following is an example of how to use the DELIMITED option of the RECEIVE command:

```
receive fileid(user01.binary.data) account(act1) userid(user05)
class(hostbin) delimited(corresponds to sender’s method);
```

MVS load modules

Expedite Base/MVS cannot send MVS load modules directly from a load library. You must convert the load module to a format supported by Expedite Base/MVS before sending it. A simple way to do this is to use the TSO XMIT command to place the load module into a sequential data set. Then, you can use Expedite Base/MVS to send the sequential data set as a binary file.

TSO XMIT command

```
xmit node.user da('user01.load(mypgm)') outdsn('user01.outpgm')
```

NOTE: If you specify an output data set that does not already exist, you must free the data set with the TSO FREE command after the XMIT command. Otherwise, you will get an error stating that the file is in use when you try to send it.

Expedite Base/MVS SEND command

```
send fileid(user01.outpgm) account(act1) userid(user04) class(loadmod)
   datatype(B) delimit(l);
```

After receiving the file with Expedite Base/MVS, your trading partner can use the TSO RECEIVE command to restore the file to load module format.

Expedite Base/MVS RECEIVE command

```
receive fileid(user04.inload) account(act1) userid(user01)
class(loadmod) delimited(L);
```

TSO RECEIVE command

```
receive indsn('user04.inload')
```

Communicating with older Information Exchange interfaces

The CDH is an important aid to file transfer. Some interfaces do not support the CDH, but Expedite Base/MVS can still communicate with these interfaces. However, there are restrictions on some features that depend on the CDH. These restrictions include the following:

- Older versions of expEDIte/PC cannot properly send or receive binary files. For more information on binary files, see “Communicating with ASCII operating systems” on page 24.
- Expedite Base/MVS ignores the AUTOEDI parameter of the RECEIVE command in files received from older interfaces.
- Expedite Base/MVS cannot format data received from older interfaces based on CDH information, because no CDH is received from older interfaces. It formats data as indicated in the DELIMITED parameter of the RECEIVE command. Expedite Base/MVS ignores the DLMOVERRIDE parameter of the RECEIVE command.
- Some of the information displayed in the RECEIVED record (for example, the DESCRIPTION parameter) is not available for files received from older interfaces.
- When you use a RECEIVE command with NONEDIONLY(Y), or a RECEIVEEDI command with EDIONLY(Y), Expedite Base/MVS does not receive files that are lacking a CDH, or that have a CDH indicating the file is not EDI data.

Any Information Exchange interface before Release 3.0 does not support the CDH.

Sending and receiving EDI data

One of the primary uses of Expedite Base/MVS is sending and receiving data formatted for electronic data interchange (EDI). Expedite Base/MVS provides a single set of commands for all EDI data transmission: SENDEDI and RECEIVEEDI. The following sections explain how these commands work.

For information on sending and receiving compressed EDI data, see Appendix E, "Using data compression."

Overview of the SENDEDI command

The SENDEDI command is used to send data formatted in X12, UCS, EDIFACT, and UN/TDI formats to Information Exchange.

A group of EDI transactions with a single destination address is an EDI envelope. Expedite Base/MVS uses several different EDI envelope definitions. The definition that is used depends on the type of EDI data the envelope contains. For details on how SENDEDI defines the envelopes, see "EDI envelopes" on page 30.

Expedite Base/MVS can transmit multiple EDI envelopes with different addresses from a single file with a single SENDEDI command. It also transmits multiple types of EDI data from a single file. You can combine X12, UCS, EDIFACT, and UN/TDI data in one file and transmit it to multiple Information Exchange destinations with one SENDEDI command.

Using the SENDEDI command, you can send data to Information Exchange without specifying an Information Exchange destination. The SENDEDI command matches EDI destinations contained in the EDI data to Information Exchange destinations.

SENDEDI determines where to send an EDI envelope by examining the contents of that envelope. The location of the destination within an EDI envelope is determined by the envelope definition (the type of EDI data being transmitted). For information on where in the envelope SENDEDI finds the envelope definition and the EDI destination, see "EDI envelopes" on page 30.

To determine an EDI envelope's destination, the SENDEDI command finds the name of the appropriate EDI destination table and centralized alias table in the EDI qualifier table. After checking the EDI qualifier table, the SENDEDI command looks for the envelope's destination in the EDI destination table. If Expedite Base/MVS finds the destination, it sends the data to the Information Exchange destination corresponding to the EDI destination. If Expedite Base/MVS does not find the destination in the EDI destination table, it uses the centralized alias table to resolve the EDI destination. For details, see “Using EDI distribution lists” on page 37.

You can send EDIFACT or X12 data to an Information Exchange destination when you know that the program cannot resolve the destination from information in the tables. Place ZZ in the qualifier ID of the EDI destination in the file you are sending. Then, you can use the Information Exchange account and user ID to specify the EDI destination. For more information, see “Bypassing tables” on page 36.

Expedite Base/MVS also enables you to send the same EDI envelope to more than one person at a time. To do this, create a distribution list and send the envelope to the list. For more information on distribution lists, see “Using EDI distribution lists” on page 37.

EDI envelopes

When you send EDI transactions, you can group the EDI transactions for a single destination within a single envelope. The EDI envelope definitions for each EDI data type are described here.

This EDI data type:	Uses this EDI envelope definition:
X12	Data between and including the ISA and IEA segments.
EDIFACT	Data between and including the UNA (or UNB) and UNZ segments.
UN/TDI	Data between and including the SCH (or STX) and END segments.
UCS	Data between and including the BG and EG segments.

The location of the destination within an EDI envelope is determined by the type of EDI data being translated. Expedite Base/MVS determines the EDI destination as described here.

This EDI data type:	Contains the EDI destination in this segment:
X12	**ISA.** SENDEDI takes the actual EDI destination from the interchange receiver ID element (ISA08). It takes the EDI qualifier from the interchange ID qualifier element (ISA07).
EDIFACT	**UNB.** SENDEDI takes the destination from data element 0010 in composite data element S003 (interchange recipient). It takes the EDI qualifier from the data element 0007 in composite data element S003 (interchange recipient).
UN/TDI	**STX.** SENDEDI takes the destination from the first subelement of the UNTO element (the recipient code address). If it does not find the recipient code, it uses the second subelement of the UNTO element (the recipient clear address) as the actual Information Exchange account and user ID.
UCS	**BG.** SENDEDI takes the EDI destination from the application receiver’s code (BG04) element.

Data between EDI segments

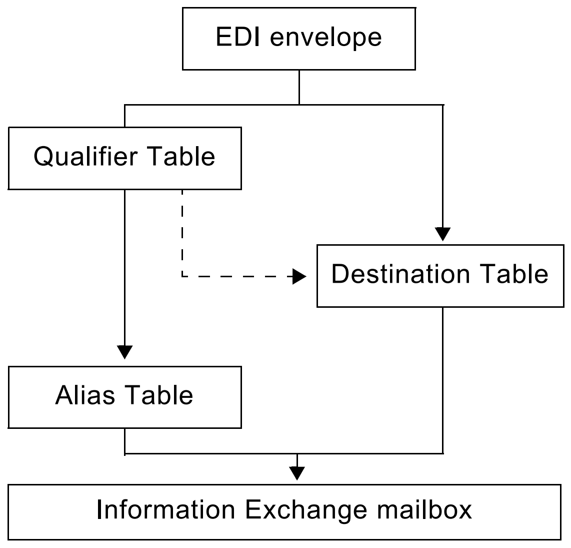
If you insert certain characters at the end of EDI segments or envelopes when preparing EDI information, Expedite Base/MVS does not consider these characters part of the EDI data. The characters Expedite Base/MVS ignores are new line ('15'x), blank ('40'x), carriage return ('0D'x), EOF ('1A'x), and tab ('05'X).

The SENDEDI command accepts the EDI data with these characters between EDI segments or envelopes, but it removes them before transmitting the data to Information Exchange. Receive commands, which receive the data as EDI (RECEIVEEDI and RECEIVE with certain options), accept the data with these characters but remove them before writing the file.

Sending EDI data

The following sections give you information on the different ways you can send EDI data using the SENDEDI command. For detailed information on the format of this command, see “SENDEDI command” on page 124.

The following flowchart illustrates how the SENDEDI command locates EDI destinations in most cases.



NOTE: SENDEDI can also locate destinations using actual Information Exchange destinations within the EDI data and distribution lists. If you plan to use Information Exchange destinations or distribution lists, see “Bypassing tables” on page 36 and “Using EDI distribution lists” on page 37 for more information.

The SENDEDI command can use three tables to determine the Information Exchange destination from the receiver ID specified in the EDI data: qualifier, destination, and alias.

If a qualifier table (QUALTBL) is provided, this is the first table SENDEDI uses. Each entry in this table indicates an EDI destination table, a centralized alias table for a given EDI qualifier and EDI data type, or both. A qualifier table is not required for SENDEDI to function.

The next table SENDEDI uses is the EDI destination table. Each entry in this table indicates an account and user ID combination, alias and alias name combination, or distribution list that corresponds to the receiver ID. You can create as many EDI destination tables as you like; each table can have as many entries as you need. No particular ddname is required for destination tables.

If you specified a centralized alias table in QUALTBL, and you either did not specify an EDI destination table or SENDEDI found no match in the EDI destination table, SENDEDI sends the EDI data using the centralized alias table as the ALIAS parameter and the receiver ID from the data as the ALIASNAME.

The SENDEDI command resolves the destination using the following procedure:

1. SENDEDI finds the receiver ID and the qualifier in the EDI data being sent. Because the UCS and UN/TDI standards do not have a qualifier, the qualifier for these two data types defaults to blanks.

2. If a qualifier table (QUALTBL) exists, SENDEDI searches the qualifier table for the first entry that matches the data type and the qualifier for the EDI envelope being processed.

   NOTE: Specifying a blank in the QUALIFIER parameter or no QUALIFIER parameter on a qualifier table entry matches all qualifiers found in the data for the matching DATATYPE. If this generic entry appears prior to a more specific entry (an entry with the QUALIFIER parameter specified with a nonblank value), the specific entry will never be used.

   If SENDEDI finds a matching entry in the qualifier table and both the EDI destination table and the centralized alias table are found on that entry, SENDEDI uses the EDI destination table first to resolve the address. If no match is found, SENDEDI uses the centralized alias table. If only one of these tables is provided, SENDEDI uses that table.

   When SENDEDI does not find a match, it leaves the centralized alias blank, and the EDI destination table ddname defaults as follows:

This EDI data type:	Defaults to this EDI destination table ddname:
X12	TTABLE*xx*, where *xx* is the 2-character qualifier.
EDIFACT	TTABLE*xx,* where *xx* is the first two characters of the qualifier.
UN/TDI	IEUNTDI.
UCS	TTABLE01.

   NOTE: A blank qualifier always defaults to account and user ID. SENDEDI never treats //TTABLExx as a match if *xx* is blank.

3. If you specify the EDI destination table, or if SENDEDI sets it by default, SENDEDI searches that table for a matching receiver ID (EDIDEST). If SENDEDI finds a matching receiver ID, it sends the EDI data to the Information Exchange destination specified in that entry. If no match is found, or if a centralized alias table was coded in the qualifier table, SENDEDI sends the EDI data using the centralized alias table as the ALIAS parameter and the receiver ID from the data as the ALIASNAME.

When SENDEDI cannot find a destination in the qualifier table or the destination table, SENDEDI determines the Information Exchange destination for each EDI data type as follows:

If the EDI data is:	SENDEDI determines the Information Exchange destination as follows:
X12, and the qualifier is ZZ or blank	The SENDEDI command splits the receiver ID into an account ID and a user ID. It uses the first 7 characters as the account ID and the last 8 characters as the user ID.
X12, and the qualifier is not ZZ or blank	This is an error, and Expedite Base/MVS does not send the data.
EDIFACT, and the qualifier, which is 0007 in composite data element S003 (interchange recipient), is ZZ or blank	SENDEDI splits the receiver code, which is 0010 in composite data element S003 (interchange recipient), into the account ID and user ID. The account ID and user ID are separated by one or more blanks, a period, or a slash.
EDIFACT, and the qualifier, which is 0007 in composite data element S003 (interchange recipient), is not ZZ or blank	This is an error, and Expedite Base/MVS does not send the data.
UN/TDI, and the recipient code (UNTO:1) is not specified	SENDEDI splits the recipient clear code (UNTO:2) into an account ID and user ID. The account ID and user ID are separated by one or more blanks, a period, or a slash.
UN/TDI, and UNTO:1 was specified	This is an error, and Expedite Base/MVS does not send the data.
UCS	This is an error, and Expedite Base/MVS does not send the data.

Using EDI qualifier tables

Expedite Base/MVS converts EDI destinations to Information Exchange destinations differently for different types of EDI data. The EDI qualifier table tells the SENDEDI command how to resolve a destination based on two pieces of information:

- The type of EDI data: X12, UCS, EDIFACT, or UN/TDI.
- The ID qualifier for a specific type of data; for example, 01 for an X12 DUNS number.

Using the above data, the EDI qualifier table provides Expedite Base/MVS with one or both of the following:

- The file or ddname of an EDI destination table that Expedite Base/MVS can use to convert the actual EDI destination to an Information Exchange destination.

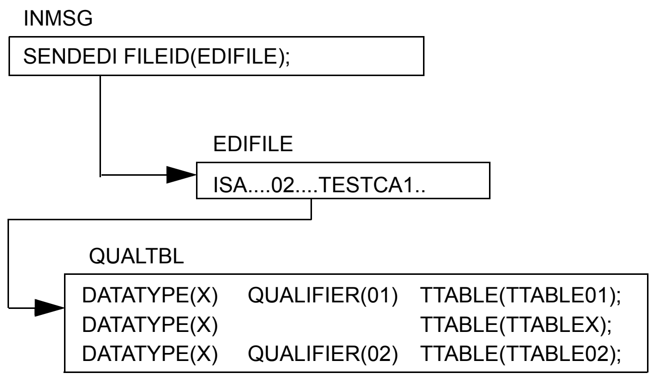
  This table is used first if both this table and the centralized alias table are provided.
- The centralized alias table name to use if Expedite Base/MVS cannot find the EDI destination in the EDI destination table.

This table is used second if both an EDI destination table and a centralized alias table are provided. This table is used first if it is the only table provided.

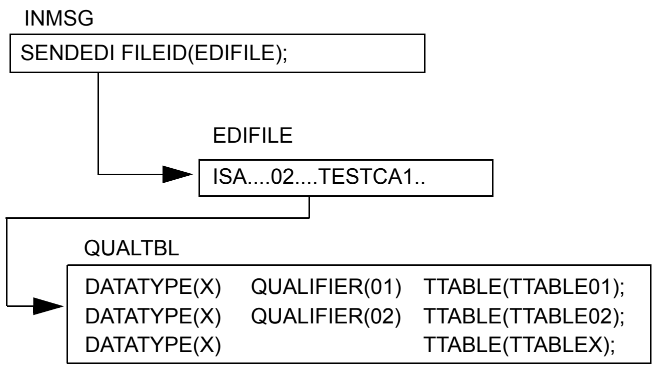
Give the ddname QUALTBL to the file in which you intend to create the EDI qualifier table. For a description of the EDI qualifier table format, see “EDI qualifier table entry format” on page 41.

NOTE: The SENDEDI command does not require the EDI qualifier table to function. If the table is not present, Expedite Base/MVS uses defaults for the EDI destination table name, as described earlier in this section.

The following figure shows the importance of order among qualifier table entries for the same data type. This example does not show the EDI destination tables.


INMSG
SENDEDI FILEID(EDIFILE);
EDIFILE
ISA....02....TESTCA1..
QUALTBL
DATATYPE(X) QUALIFIER(01) TTABLE(TTABLE01);
DATATYPE(X) TTABLE(TTABLEX);
DATATYPE(X) QUALIFIER(02) TTABLE(TTABLE02);

Expedite Base/MVS uses TTABLEX to resolve the destination TESTCA1 because its qualifier table entry is read before the more explicit qualifier table entry specifying TTABLE02. If TTABLE02 should have been used, reorder the entries in the qualifier table as shown.


INMSG
SENDEDI FILEID(EDIFILE);
EDIFILE
ISA....02....TESTCA1..
QUALTBL
DATATYPE(X) QUALIFIER(01) TTABLE(TTABLE01);
DATATYPE(X) QUALIFIER(02) TTABLE(TTABLE02);
DATATYPE(X) TTABLE(TTABLEX);

Using EDI destination tables

Think of an EDI destination table as a list of EDI destinations paired with Information Exchange destinations. The SENDEDI command resolves destinations by searching for an EDI destination and then using the corresponding Information Exchange destination as the actual address for an envelope. For a description of the EDI destination table format, see “EDI destination table entry format” on page 42.

To send EDI data to a destination defined in an EDI destination table, follow these steps:

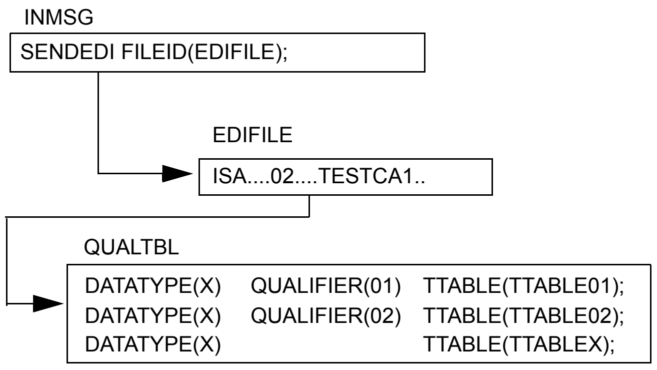
1. Build an EDI destination table that contains your EDI destination and the corresponding Information Exchange destination.

2. Build an EDI qualifier table that contains the name of your EDI destination table and the type of EDI data you want to send.

   NOTE: If you use the default name for your EDI destination table, you do not have to build an EDI qualifier table.

3. Use the SENDEDI command to send the EDI envelope.

The following figure shows the tables used to send an X12 file to a trading partner with an EDI destination of TESTDUN1 and a qualifier of 01.


INMSG
SENDEDI FILEID(EDIFILE);
EDIFILE
ISA....02....TESTCA1..
QUALTBL
DATATYPE(X) QUALIFIER(01) TTABLE(TTABLE01);
DATATYPE(X) QUALIFIER(02) TTABLE(TTABLE02);
DATATYPE(X) TTABLE(TTABLEX);

Expedite Base/MVS sends the data in the example above to a destination with an Information Exchange account ID of IEACCT1 and a user ID of IEUSER1.

Using centralized Information Exchange alias tables

You might find it time consuming to maintain EDI destination tables in multiple locations. To avoid this problem, the SENDEDI command allows you to use centralized Information Exchange alias tables. These permanent tables reside within Information Exchange and can convert EDI destinations into Information Exchange destinations. You can make them available to all Information Exchange users (global alias tables), members of a particular account (organization alias tables), or a single user (private alias tables). The EDI qualifier table and the EDI destination table determine which, if any, of these alias tables should be used.

NOTE: There are two ways for you to create and maintain alias tables:

- Using Information Exchange Administration Services (see Using Information Exchange Administration Services).
- Using the DEFINEALIAS command (see “DEFINEALIAS command” on page 86).

To send EDI data to a destination defined in an Information Exchange centralized alias table, follow these steps:

1. Add the target EDI destination to a centralized alias table using Information Exchange Administration Services.

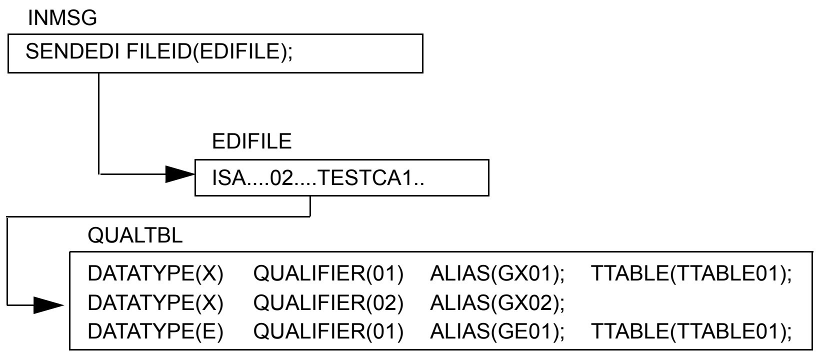
2. Build an EDI qualifier table that contains the name of the Information Exchange centralized alias table and the type of EDI data you want to send.

NOTE: Expedite Base/MVS contains a sample EDI qualifier table. This table defines standardized central alias tables for all types of EDI data. In some Information Exchange installations, the U.S., for example, these standardized, centralized alias tables have already been created. You can add your EDI destinations to these tables as needed to resolve your destinations.

If you use an EDI destination table, be sure that your EDI destination table does not contain the name of the target destination.

3. Use the SENDEDI command to send the file containing your EDI data.

The following figure shows the tables used to send an X12 file to a trading partner with an EDI destination of TESTSCA1 and a qualifier of 02. This particular example does not include an EDI destination table.


INMSG
SENDEDI FILEID(EDIFILE);
EDIFILE
ISA....02....TESTCA1..
QUALTBL
DATATYPE(X) QUALIFIER(01) ALIAS(GX01); TTABLE(TTABLE01);
DATATYPE(X) QUALIFIER(02) ALIAS(GX02);
DATATYPE(E) QUALIFIER(01) ALIAS(GE01); TTABLE(TTABLE01);

The data in the example above is sent to an Information Exchange destination with the Information Exchange alias table GX02 and an alias name of TESTSCA1.

Bypassing tables

If you need to send only EDIFACT, X12, or UN/TDI data to an Information Exchange destination, you can bypass the tables and send the EDI information directly to an Information Exchange destination.

To send EDIFACT or X12 data to an Information Exchange destination contained within the EDI data, follow these steps:

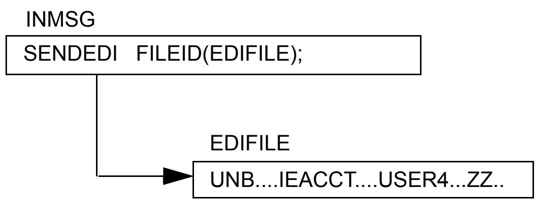
1. Place **ZZ** in the qualifier.
2. Use the Information Exchange account ID and user ID as the actual EDI destination. When sending X12 data, separate the account ID and user ID with at least one blank. When sending EDIFACT data, separate the account ID and user ID with a period, a slash, or a blank.
3. Use the SENDEDI command to send the file that contains your EDI data.

NOTE: When you use a ZZ qualifier, Expedite Base/MVS tries to resolve the destination by searching the tables. When it does not locate the destination in the tables, it sends the data to the Information Exchange account and user ID specified. If you do not want Expedite Base/MVS to refer to the tables first, use a blank qualifier instead of a ZZ qualifier.

To send UN/TDI data to an Information Exchange destination contained within EDI data, follow these steps:

1. Do not specify the recipient code (UNTO:1).
2. Place the Information Exchange account and user ID in the recipient clear subelement (UNTO:2).
3. Use the SENDEDI command to send the file that contains your EDI data.

The following figure shows the route that Expedite Base/MVS uses to send an EDIFACT file to a trading partner with an Information Exchange destination of IEACCT USER4.



Expedite Base/MVS sends the data in the example above to an Information Exchange destination with an Information Exchange account ID of IEACCT and a user ID of USER4.

Intersystem addressing for EDIFACT and UN/TDI

Intersystem addressing is available with EDIFACT and UN/TDI data when using the SENDEDI command.

Instead of using an alias table to address EDI files, you can place an Information Exchange address in the EDIFACT or UN/TDI header specifying the appropriate IDs in the following order:

1. System ID (optional if the sender and receiver are using the same system)
2. Account ID
3. User ID

All of the above IDs must be separated by one of the following:

- Period (.)
- Slash (/)
- One or more blank spaces

For EDIFACT tables, SENDEDI splits the receiver code, which is data element 0010 in composite data element S003 (Interchange Recipient), into the system, account, and user IDs.

For UN/TDI data, SENDEDI splits the recipient clear code (UNTO:2) into the system, account, and user IDs.

Using EDI distribution lists

To send EDI data to an Information Exchange list, follow these steps:

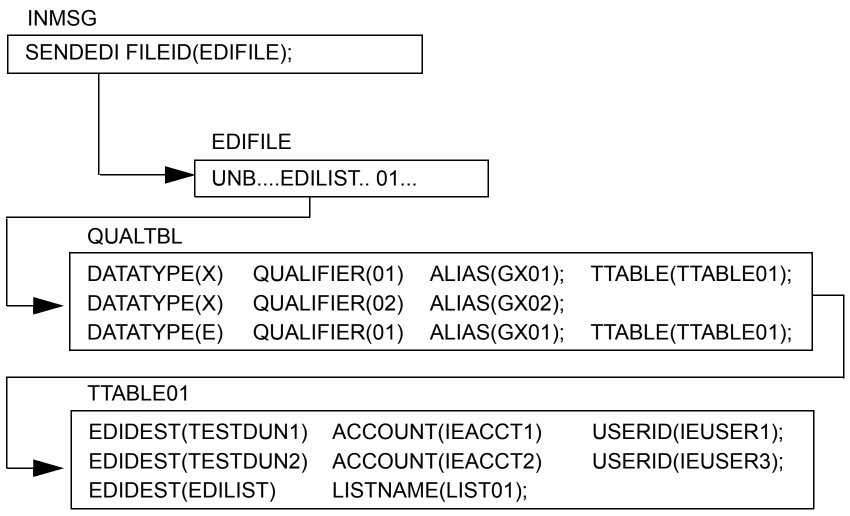
1. Define the Information Exchange list.

   You can define a list using Expedite Base/MVS or Information Exchange Administration Services.

2. Build an EDI destination table that contains your EDI destination and the corresponding Information Exchange list name.
3. Build an EDI qualifier table that contains the name of your EDI destination table and the type of EDI data it references.

   NOTE: If you use a default name for your EDI destination table, you do not have to build an EDI qualifier table.
4. Use the SENDEDI command to send the file that contains your EDI data.

The following figure shows the tables used to send an EDIFACT file to a list of trading partners with an EDI destination of EDILIST and a qualifier of 01.


INMSG
SENDEDI FILEID(EDIFILE);
EDIFILE
UNB....EDILIST.. 01...
QUALTBL
DATATYPE(X) QUALIFIER(01) ALIAS(GX01); TTABLE(TTABLE01);
DATATYPE(X) QUALIFIER(02) ALIAS(GX02);
DATATYPE(E) QUALIFIER(01) ALIAS(GX01); TTABLE(TTABLE01);
TTABLE01
EDIDEST(TESTDUN1) ACCOUNT(IEACCT1) USERID(IEUSER1);
EDIDEST(TESTDUN2) ACCOUNT(IEACCT2) USERID(IEUSER3);
EDIDEST(EDILIST) LISTNAME(LIST01);

Expedite Base/MVS sends the data in the example above to an Information Exchange destination with a list name of LIST01.

To create a permanent list and store it in Information Exchange, use the LIST command described in “LIST command” on page 93.

You can also use Information Exchange Administration Services to create a permanent list. For more information on using Information Exchange Administration Services to create and update this type of list, refer to the *Information Exchange Administration Services User’s Guide*.

Specifying Information Exchange control fields

You can specify the following Information Exchange control fields by using the parameters within the SENDEDI command.

Message name (MSGNAME) assignment

If you provide a MSGNAME parameter when using the SENDEDI command, SENDEDI uses this value for the Information Exchange message name. If you do not provide the MSGNAME parameter, the SENDEDI command generates the Information Exchange message name based on the type of EDI data transmitted. The following table describes how the SENDEDI command generates the message name:

This EDI data type:	Generates this MSGNAME:
EDIFACT data	Expedite Base/MVS takes the MSGNAME from the data element 0020 (Interchange Control Reference) of the EDI data. If the element exceeds 8 bytes in length, Expedite Base/MVS uses the first 8 bytes. If the element is fewer than 8 bytes in length, Expedite Base/MVS places it in the MSGNAME parameter value. Left-justified and padded with blanks.
UN/TDI data	Expedite Base/MVS takes the MSGNAME from the sender's reference field (SNRF) of the EDI data. If the SNRF exceeds 8 bytes in length, Expedite Base/MVS uses the first 8 bytes. If the SNRF is fewer than 8 bytes in length, Expedite Base/MVS places it in the MSGNAME parameter value. Left-justified and padded with blanks.
X12 data	Expedite Base/MVS takes the MSGNAME from the last 8 bytes of the interchange control number of the X12 data.
UCS data	Expedite Base/MVS takes the MSGNAME from the interchange control number. Because the UCS interchange control number has a maximum length of 5 bytes, Expedite Base/MVS places the interchange control number in the MSGNAME parameter value. Left-justified and padded with blanks.

Message sequence number (MSGSEQNO) assignment

If you provide a MSGSEQNO parameter in the SENDEDI command, the SENDEDI command uses this value for the Information Exchange message sequence number. If you do not provide the MSGSEQNO parameter, the SENDEDI command generates the Information Exchange message sequence number in the following way for all EDI data types.

The SENDEDI command counts each EDI envelope transmitted from a single data file. It places the count in the MSGSEQNO of the Information Exchange messages corresponding to the EDI envelopes. SENDEDI formats MSGSEQNO as a series of numeric characters ranging from 00001 to 99999. Therefore, three EDI envelopes sent from a single file would have the following MSGSEQNO values:

- 00001 for the first EDI envelope in the file
- 00002 for the second EDI envelope in the file
- 00003 for the last EDI envelope in the file

Each time you use the SENDEDI command for a new file that contains EDI envelopes, or when the MSGSEQNO reaches 99999, the MSGSEQNO counter resets to 00001.

Message class (CLASS) assignment

If the SENDEDI command CLASS parameter is not blank, the SENDEDI command uses this value. If you do not provide the CLASS parameter, the SENDEDI command generates the parameter based on the type of EDI data transmitted.

EDIFACT and UN/TDI data

For EDIFACT and UN/TDI data, Expedite Base/MVS takes the class from the Application reference field (APRF) of the EDI data. If the APRF exceeds 8 bytes in length, Expedite Base/MVS uses the first 8 bytes. If the APRF is fewer than 8 bytes in length, Expedite Base/MVS places the APRF in the Class field, left-justified and padded with blanks. If the APRF is not present, Expedite Base/MVS sets the CLASS parameter as follows:

This EDI data type:	Defaults to this class:
EDIFACT	#EE
UN/TDI	#EU

X12 and UCS data

For X12 and UCS data, Expedite Base/MVS sets the CLASS parameter as follows:

This EDI data type:	Defaults to this class:
X12	#E2
UCS	#EC

SENDEDI response records

The SENT record provides a record of the EDI envelopes sent by the SENDEDI command. For a description of the format of this record, see “SENDEDI command” on page 124.

The NOTSENT record provides a record of the EDI envelopes not sent by the SENDEDI command due to a destination verification failure. NOTSENT records are given only when VERIFY (c or g) is specified. For a description of the format of this record, see “NOTSENT record” on page 163.

NOTE: You can use the SENT records in the message response file (OUTMSG) to determine which EDI envelopes have been sent when the SENDEDI command does not complete, and you can use the NOTSENT records to determine which envelopes were not sent.

Receiving EDI data

You use the RECEIVEEDI command to receive EDI data from Information Exchange. You can receive multiple EDI envelopes that contain different types of data with a single RECEIVEEDI command.

The RECEIVEEDI command is similar to the RECEIVE command, but the ability to reformat received data based on EDI segment delimiters makes it different from the RECEIVE command. The EDIOPT parameter of the RECEIVEEDI command controls this function. For more information on the format of this command, see “RECEIVEEDI command” on page 111.

If your trading partner sends you EDI data with a CDH, the RECEIVE command recognizes the EDI format of the data and reformats it accordingly. The AUTOEDI parameter of the RECEIVE command controls this function. Expedite Base/MVS always prepares a CDH when sending data. Information Exchange interfaces prior to Release 3.0 do not support the CDH.

Using the RECEIVEEDI command does not guarantee that you receive only EDI data. Use the CLASS parameter to select only the class of messages you want to receive.

If the person sending you data uses the default user classes of the SENDEDI command, you can use the Information Exchange wildcard receive feature to simplify receipt. For example, by specifying **#E?** as the user class in the RECEIVEEDI command, you ask Information Exchange to return only files that have a user class beginning with #E. This includes all files sent with the default EDI user classes.

NOTE: The SENDEDI command places a single EDI envelope in an Information Exchange message group. The RECEIVEEDI command expects each EDI envelope to be contained in a separate Information Exchange message group. If you put multiple EDI envelopes in a single file and send them using the SEND (not the SENDEDI) command, the results are unreliable.

If the CDH indicates the data is not EDI, the RECEIVEEDI command does not perform any special EDI processing.

If there is no CDH, RECEIVEEDI attempts to process the files as EDI data. If this is not possible, Expedite Base/MVS writes the data without reformatting.

EDI qualifier table entry format

The EDI qualifier table resides in the file referenced by the ddname QUALTBL. Each entry in this table indicates an EDI destination table, a centralized alias table for a given EDI qualifier or EDI data type, or both.

Format

```
datatype(data type) qualifier(qual) ttable(ttable) alias(alias);
```

Parameters

datatype
Indicates the EDI data type for this entry.

blank	For all EDI data types. This entry matches any of the supported EDI data types. This is the default.
x	For X12 data
c	For UCS data
e	For EDIFACT data
u	For UN/TDI data

qualifier
Indicates the EDI qualifier for this entry. If this parameter is blank, the entry matches any EDI qualifier. Use 1 to 4 alphanumeric characters. The default is blank.

ttable
Indicates the name of the EDI destination table. If you specify a table in the TTABLE parameter, SENDEDI uses that table to try to resolve the Information Exchange destination. If you do not specify a destination table, SENDEDI does not use a table for EDI data matching this entry. If this parameter refers to a ddname, begin the parameter with dd:. Otherwise, the parameter value is a file name.

For a ddname, follow dd: with 1 to 8 alphanumeric characters. For a file name, use 1 to 54 alphanumeric characters.

alias
Indicates the centralized alias table for this entry. If you specify this parameter and SENDEDI does not find the destination in the EDI destination table, SENDEDI uses this alias table with the EDI receiver ID as the alias name.

blank	No centralized alias table is used. This is the default.
G*xxx*	Global alias table, where *xxx* identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

You can use a “#” (number-sign) symbol to define comments. Type any information after #. The program ignores the characters after # until it encounters a new line. You do not have to end comments with #. Expedite Base/MVS considers the end of the line in which the comment exists to be the end of that comment. If you want the comment to continue, begin a new line with #.

EDI destination table entry format

The EDI destination table resides in a file specified by the TTABLE parameter in the EDI qualifier table. Each entry in this table indicates the Information Exchange destination associated with a given EDI receiver ID.

Format

```
edidest(edidest)

alias(alias) aliasname(aliasname);
   or
sysid(sysid) account(account) userid(userid);
   or
account(account) userid(userid);
   or
listname(listname);
```

Parameters

edidest
Indicates the EDI receiver ID for this entry. If this parameter matches the receiver ID from the EDI data, Expedite Base/MVS sends the message to the Information Exchange destination specified by the parameters you select. Use 1 to 35 alphanumeric characters.

alias
Indicates the alias table type and table name.

blank	No alias table name is used.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

aliasname
Indicates an alias name defined in the alias table. Use 1 to 16 alphanumeric characters.

sysid
Indicates the system ID of a single-destination user ID. You need this only if the specified account ID and user ID reside on another Information Exchange system. You can use it only with the ACCOUNT and USERID parameters. Use 1 to 3 alphanumeric characters.

account
Indicates the account name of a single-destination user ID. Use 1 to 8 alphanumeric characters.

userid
Indicates the destination user ID. Use 1 to 8 alphanumeric characters.

listname
Indicates the name of a previously defined list of account IDs and user IDs. Use 1 to 8 alphanumeric characters.

You can use a # (number sign) symbol to define comments. Type any information after #. The program ignores the characters after # until it encounters a new line. You do not have to end comments with #. Expedite Base/MVS considers the end of the line in which the comment exists to be the end of that comment. If you want the comment to continue, begin a new line with #.

Chapter 5

Using checkpoint-level, file-level, and user-initiated recovery

This chapter explains in detail how to use the checkpoint-level, file-level, and user-initiated data recovery features of Information Exchange. It provides information on resuming or resetting an existing Information Exchange session.

When you use session-level recovery (described in the previous chapter), Expedite Base/MVS must retransmit all data after an error. If you are sending a large amount of data, retransmission can take several hours.

When you use checkpoint-level, file-level, or user-initiated recovery, Expedite Base/MVS can recover data more efficiently. The following table shows when Expedite Base/MVS takes checkpoints for each of these recovery methods.

Checkpoint-level recovery is usually the best way to exchange data when transferring a large number of files or a large amount of data.

Checkpoint-level recovery	File-level recovery	User-initiated recovery
• After sending the number of bytes you specify in the COMMITDATA parameter of the TRANSMIT command (default is 141000 bytes) • At the end of each SEND, SENDEDI, or PUTMEMBER command, if the next command is not a SEND, SENDEDI, or PUTMEMBER command • While receiving data for a RECEIVE or RECEIVEEDI command, if the file was sent with checkpoints • At the end of each RECEIVE or RECEIVEEDI command • Recovery ensures that if the session is discon-nected, Expedite can pick up the data transfer at the last checkpoint rather than start the transfer over from the beginning	• After each file sent as a result of a SEND, SENDEDI, or PUTMEMBER command • After each file is received • While receiving data for a RECEIVE or RECEIVEEDI command, if the file was sent with checkpoints • At the end of each RECEIVE or RECEIVEEDI command	• After each COMMIT command, unless there is nothing to commit • At the end of each session, even if you have not specified a COMMIT command • While receiving data for a RECEIVE or RECEIVEEDI command, if the file was sent with checkpoints • At the end of each RECEIVE or RECEIVEEDI command **NOTE:** If the INMSG data set is a SYSIN data set, then a commit is done after each SEND, SENDEDI, or PUTMEMBER command.

The processes for using checkpoint-level, file-level, and user-initiated recovery are very similar. Expedite Base/MVS uses the same work files for these recovery methods. Considerations for restarting after an error and resetting the Expedite Base/MVS session, described later in this chapter, also apply to all three recovery methods.

NOTE: SENDSTREAM and RECEIVESTREAM are not valid for these recovery methods.

Selecting checkpoint-level, file-level, or user-initiated recovery

Session-level recovery is the default in Expedite Base/MVS. Request one of the other recovery methods by specifying one of these values on the RECOVERY parameter of the TRANSMIT command:

c	Checkpoint-level recovery
f	File-level recovery
u	User-initiated recovery

Recovery considerations

Checkpoint-level, file-level, and user-initiated recovery are easy to use if you allow Expedite Base/MVS to successfully complete its current session before processing the OUTMSG file or making any new requests.

ATTENTION: Do not use the same account and user ID on multiple systems. If you start an Information Exchange session using checkpoint-level recovery, file-level recovery, or user-initiated recovery while another Information Exchange session with the same account and user ID is running, Information Exchange ends the first session and continues the second session. The results in the first session depend on whether a checkpoint ended successfully:

- If a checkpoint ended successfully, Information Exchange delivers any data sent prior to the checkpoint and deletes any data from the mailbox that was received prior to the checkpoint.
- If a checkpoint did not end successfully, Information Exchange does not deliver any data and does not delete any received data from the mailbox. This means that data received in the first session may be received again in error.

In either case, you may get an error when you restart the first session. You should not run multiple sessions for the same account and user ID from different machines. Also, you should not share work files between different jobs.

Setting up work files

To take checkpoints, Expedite Base/MVS must store the status of your Information Exchange session in a safe place. It must also close and reopen your message response file (OUTMSG) when it completes a checkpoint. For these reasons, Expedite Base/MVS requires that you create extra work files, SESSION, RCVWORK, and OUTWORK, when using one of these recovery methods.

When you use the SENDEDI command with VERIFY(C) or VERIFY(G) specified, you must also create an EDIWORK file. Each of these files must have the following characteristics:

- It must be a sequential file on DASD. PDS members or non-DASD device types are not permitted.
- It must be a permanent file, not a temporary or SYSOUT data set.
- It must not be a DISP=MOD file.
- It must be contained in one DASD volume.
- It must not be a VSAM file.
- It can use a fixed or variable record format (not undefined).
- It may be allocated with any valid record length. Expedite Base/MVS will adjust the record length and record format to the values it needs.

NOTE: Do not code the RLSE parameter in your JCL or *all* space except the minimum amount defined is released.

The following sections provide more details on how to create each work file.

Creating the session file (SESSION)

Expedite Base/MVS uses the session file to record the status of your Information Exchange session when it takes a checkpoint.

Expedite Base/MVS does not create the session file. You must create it as a permanent file and refer to it with the ddname SESSION when you run Expedite Base/MVS. The session file must be at least 35000 bytes long.

You should not erase or modify the session file after an unsuccessful Expedite Base/MVS session. If you erase or modify the session file, Expedite Base/MVS reprocesses all of the requests in your INMSG file. If you have already received files, Expedite Base/MVS overwrites them. If you have already sent files, Expedite Base/MVS resends them.

Creating the receive work file (RCVWORK)

When you use checkpoint-level, file-level, or user-initiated recovery, Expedite Base/MVS receives data into a work file. After it receives all the data, Expedite Base/MVS moves it from the work file into the file specified with the FILEID parameter of the RECEIVE or RECEIVEEDI command. At this point, RCVWORK is ready to receive the next transmission.

Expedite Base/MVS does not create the receive work file. You must create it as a permanent file and refer to it with the ddname RCVWORK when you run Expedite Base/MVS. Make the receive work file large enough to hold all of the data received by any single RECEIVE or RECEIVEEDI command in your message command file. For example, if you specify one RECEIVE command for which you expect to receive 5000 bytes of data, and another for which you expect to receive 5,000,000 bytes of data, you must make your receive work file large enough to hold 5,000,000 bytes of data.

Creating the EDI work file (EDIWORK)

When you use the SENDEDI command with the VERIFY parameter equal to either c or g, Expedite Base/MVS uses the EDI work file to track which EDI envelopes are sent and which are not.

Expedite Base/MVS does not create the EDI work file. You must create it as a permanent file and refer to it with the ddname EDIWORK when you run Expedite Base/MVS. The EDI work file must be able to hold a minimum of 10,000 bytes/records or at least as many characters as the maximum number of envelopes in the EDI file being sent.

You should not erase or modify the EDI work file after an unsuccessful Expedite Base/MVS session. If you erase or modify the EDI work file, Expedite Base/MVS might send duplicate data.

Creating the output work file (OUTWORK)

When you use one of these recovery methods, Expedite Base/MVS does not put echoed commands and response records directly into the message response file. First, it places echoed commands and response records into an output work file, then it copies them to the message response file when taking a checkpoint.

Expedite Base/MVS does not create the output work file. You must create a permanent file and refer to it with the ddname OUTWORK when you run Expedite Base/MVS. Create this file with the same DCB values and the same size as your message response file, OUTMSG.

Message response file (OUTMSG) considerations

When using one of these recovery methods, Expedite Base/MVS must open and close the message response file after completing a checkpoint. For this reason, the message response file must have the following characteristics in addition to those described in "OUTMSG attributes" on page 16:

- It must not be a member of a PDS.
- It must be a permanent file, not a temporary or SYSOUT data set.

Message command file (INMSG) considerations

If you send large numbers of small files, you might experience performance problems if INMSG is a SYSIN data set or if it is a concatenation of several data sets.

Processing the message response file

Processing the message response file for these recovery methods is very much like processing for session-level recovery; however, there are differences in the way you handle errors.

Examining response records

Response records are free-format records with the same syntax as commands. For a description of command syntax, see "Understanding command syntax" on page 4. Response records always start at the beginning of a line. However, parameters in response records can occur in any position in a record and in any order. In addition, all parameters in a response record might not appear.

When examining response records, consider the following:

- Assume a default value if you do not get the response record parameter you are expecting. This is not an error.
- If a response record parameter has a longer length than you expect, truncate the parameter.
- Be prepared to handle parameters that are split across records. Splitting can occur if the parameter is longer than the record length of your response file.

Checking return codes

To ensure that Expedite Base/MVS finished processing the message command file, check the return code in the SESSIONEND record.

The following table lists the return codes you can receive, explains them, and tells you what to do when you receive them:

If the SESSIONEND return code is:	Then:	You must do the following:
00000	Expedite Base/MVS finished processing the command file. There were no processing problems, and all of the command RETURN records also contain 0 return codes. All of your requests completed normally.	Nothing.
28010	The Information Exchange session completed normally, but Expedite Base/MVS could not complete all the commands in your message command file.	Check the command RETURN records to see which commands did not complete. Those that completed normally have 0 RETURN records. Those that did not complete have nonzero RETURN records, and if your system is set up to do so, can display error descriptions.
Anything other than 00000 or 28010	Expedite Base/MVS did not finish processing the command file. Your Information Exchange session failed. However, some of the commands in your command file may have completed. Expedite Base/MVS might have placed mail in your trading partner's mailbox or removed mail from your mailbox.	Correct the problem and restart Expedite Base/MVS. The SESSIONEND record might include an error description to help you find the problem, if your system is set up to display error descriptions. A command RETURN record might contain the same code and description. If there are no return records in OUTMSG, check OUTPRO and OUTWORK for the error.

If the error was due to a problem with a command, you can determine which command is wrong by looking in one of three places:

- **OUTPRO** - If the error was in a profile command, the command in error appears in OUTPRO. In this case, OUTWORK and OUTMSG do not contain any echoed commands.
- **OUTWORK** - If no checkpoint was taken while the command was processing, the command appears in OUTWORK. The command in error is followed by a RETURN record with the same return code as the SESSIONEND record. All syntax errors appear in OUTWORK.
- **OUTMSG** - If a checkpoint was taken while the command was processing, Expedite Base/MVS displays the command in error in OUTMSG. The command is not followed by a RETURN record. However, Expedite Base/MVS echoes the command in error as the last command in OUTMSG before SESSIONEND.

Recovery response file examples

The SESSIONEND(00000) and SESSIONEND(28010) examples in “OUTMSG examples” on page 60 are valid for all data recovery methods. However, the following examples pertain only to checkpoint-level, file-level, and user-initiated recovery.

Response file with a SESSIONEND (not 00000 or 28010)

Example 1

In the following example, you see the result of a typographical error. The programmer intends to enter **sendedi**, but types **sendedk** by mistake. This results in a 15040 error in the SESSIONEND record and another 15040 error in the RETURN record following **sendedk**. Because this is a syntax error, the error message is found in OUTWORK.

INMSG

```
#Send a file named user01.test.data to act1 user01 in user class tdata#
SEND FILEID(USER01.TEST.DATA(EDI001)) ACCOUNT(ACT1) USERID(USER01)
  CLASS(TDATA);
#send a file#
SENDEDK FILEID(USER01.TEST.DATA(EDI002));
```

OUTMSG

```
AUTOSTART SESSIONKEY(USZTZDG0);
STARTED LASTSESS(0) RESPCODE(00000) SESSIONKEY(USZTZDG0) IEVERSION(04)
  IERELEASE(07);
RETURN(00000);

#Send a file named user01.test.data to act1 user01 in user class tdata#

SEND FILEID(USER01.TEST.DATA) ACCOUNT(ACT1) USERID(USER01)
  CLASS(TDATA);
SENT UNIQUEID(USZTZDG0) LENGTH(320);
RETURN(00000);
SESSIONEND(15040) ERRDESC(Command not recognized.)
ERRTEXT(EXPLANATION: You specified an unrecognized command in the
  command file.)
ERRTEXT(USER RESPONSE: Check the message response file, OUTMSG,
  profile)
ERRTEXT(response file, OUTPRO, or response work file, OUTWORK, to)
ERRTEXT(determine which command produced the error. Correct the
  appropriate)
ERRTEXT(command file and retry the program.);
```

OUTWORK

```
#Send a file#
SENDEDK
RETURN(15040) ERRDESC(Command not recognized.)
ERRTEXT(EXPLANATION: You specified an unrecognized command in the
  command file.)
ERRTEXT(USER RESPONSE: Check the message response file, OUTMSG,
  profile)
ERRTEXT(response file, OUTPRO, or response work file, OUTWORK, to)
ERRTEXT(determine which command produced the error. Correct the
  appropriate)
ERRTEXT(command file and retry the program.);
```

Example 2

This example illustrates a session using user-initiated recovery. The session does not end successfully because of a syntax error on the RECEIVE command. The RECEIVE command was entered as **recve**. The following example shows the command file,

INMSG

```
SEND FILEID(USER01.TEST.DATA(EDI0001)) ACCOUNT(ACT1) USERID(USER01);
RECEIVE FILEID(DD:FILE1);
SENDEDI FILEID(USER01.TEST.DATA(FILE0001));
COMMIT;
SEND FILEID(USER01.TEST.DATA(FILE0002)) ACCOUNT(ACT1) USERID(USER01);
RECVE FILEID(DD:FILE2);
COMMIT;
SEND FILEID(USER01.TEST.DATA(FILE0003)) ACCOUNT(ACT1) USERID(USER01);
```

The session ends with a 15040 error on **recve** and a 15040 error on the SESSIONEND record. Because this is a syntax error, the command in error is found in the OUTWORK file. Postprocessing of the response file shows which commands completed successfully. The following examples show the OUTMSG message response file and the OUTWORK response work file:

OUTMSG

```
AUTOSTART SESSIONKEY(USZUA0TL);
STARTED LASTSESS(0) RESPCODE(00000) SESSIONKEY(USZUA0TL) IEVERSION(04)
  IERELEASE(07);
RETURN(00000);
SEND FILEID(USER01.TEST.DATA(EDI0001)) ACCOUNT(ACT1) USERID(USERO1);
SENT UNIQUEID(USZUA0WF) LENGTH(320);
RETURN(00000);
RECEIVE FILEID(DD:FILE1);
RECEIVED ACCOUNT(ACT1) USERID(USER01) CHARGE(1) LENGTH(320)
  FILEID(DD:FILE1) MSGDATE(040510) MSGDATELONG(20040510) MSGTIME(152321)
  MSGSEQO(000001) SESSIONKEY(USZUA0TL) DELIMITED(N) SYSNAME(EXMV000T)
  SYSLEVEL(460) TIMEZONE(L) DATATYPE(E) EDITYPE(UNFORMATTED)
  SENDERFILE(IBM2EDI.EBMVS46T.EDIDATA(EDI001)) SENDLOC(3390    /TSO013)
  FILEDATE(040510) FILEDATELONG(20040510) RECFM(FB) RECLEN(80) RECDLM(N)
  UNIQUEID(USZUA0WF) SYSTYPE(21)SYSVER(4);
RETURN(00000);
SENDEDI FILEID(USER01.TEST.DATA(FILE0001));
COMMIT;
RETURN(00000);
SEND FILEID(USER01.TEST.DATA(FILE0002)) ACCOUNT(ACT1) USERID(USER01);
SENT UNIQUEID(USZUA17R) LENGTH(320);
RETURN(00000);
SESSIONEND(15040) ERRDESC(Command not recognized.)
ERRTEXT(EXPLANATION: You specified an unrecognized command in the)
ERRTEXT(command file.)
ERRTEXT(USER RESPONSE: Check the message response file, OUTMSG,)
ERRTEXT(profile response file,OUTPRO, or response work file,)
ERRTEXT(OUTWORK, to determine which command produced the error.)
ERRTEXT(Correct the appropriate command file and retry the program.)
```

OUTWORK

```
RECVE
RETURN(15040) ERRDESC(Command not recognized.)
ERRTEXT(EXPLANATION: You specified an unrecognized command in the)
ERRTEXT(command file.)
ERRTEXT(USER RESPONSE: Check the message response file, OUTMSG,)
ERRTEXT(profile response file, OUTPRO, or response work file,)
ERRTEXT(OUTWORK, to determine which command produced the error.)
ERRTEXT(Correct the appropriate command file and retry the program.);
```

The RETURN(00000) records indicate that the first three commands before the COMMIT command completed successfully. The RETURN(00000) and SENT records for the second SEND command indicate that this command completed successfully. However, because the second COMMIT command was not processed due to the syntax error, Information Exchange will not deliver the file for the second SEND command.

The **recve** should be corrected in the INMSG file and the session restarted. Expedite will continue processing after the last successful COMMIT command, which in this example is after the second SEND command.

Restarting after an error

Certain conditions can cause Expedite Base/MVS to stop running without completing the Information Exchange session. Examples of these conditions include communication failures and syntax errors in your message command file. When one of these conditions occurs, correct the problem that caused the condition, then restart Expedite Base/MVS by resubmitting your job. Expedite Base/MVS then uses the information contained in its session file (SESSION) to resume the previous session at the last checkpoint. If the session file is not present, Expedite Base/MVS cannot resume the session. Instead, it reprocesses all the requests in your message command file (INMSG).

CAUTION: Do not erase or alter the session file at any time. Damaging the session file during a restart situation can result in duplicated or lost data. Also, do not alter or erase the message response file, the receive work file, or the EDI work file during a restart situation. You can, however, alter the message command file to the extent necessary to correct any syntax or command error that caused the processing to stop. Never add or remove commands from the message command file between restarts.

Changing files on restart

When you restart Expedite Base/MVS from the last checkpoint, Expedite Base/MVS uses the command and response files from the session that ended in an error. In addition, Expedite Base/MVS uses several control files to keep track of its status.

CAUTION: Avoid changing any of these files when possible. Changes are sometimes required (for example, in the case of a syntax error), but be aware that improper changes can prevent Expedite Base/MVS from restarting correctly and can cause the resulting data to be unreliable.

The following list shows which files you can change before restarting and which you cannot change. Feel free to change any file that is not on the list.

- **Session file (SESSION).** Never change this file before restarting. This is a control file used by Expedite Base/MVS to restart the session.

- **Files being sent or received.** Do not modify files while sending or receiving them. Changes make the data that is sent or received unreliable.
- **Message command file (INMSG).** Do not change any commands or parameters in INMSG if they have already echoed to the message response file.

Changing INMSG commands or parameters that have already been processed and written to the message response file results in an error when you restart the program. You can change commands and parameters that have been written only to the output work file (OUTWORK) or that are not contained in OUTWORK or the message response file (OUTMSG).

- **Message response file (OUTMSG).** Do not change the message response file. Because it is a record of events that have already occurred, there is no reason to change it and doing so can corrupt data.
- **Profile command file (INPRO).** Modify the profile command file if you want to change a profile value. However, do not change the RECOVERY, MSGSIZE, COMMITDATA, or COMMTYPE parameters in the TRANSMIT command. Each time you restart, Expedite Base/MVS processes the profile commands.
- **Profile response file (OUTPRO)**. Do not change the profile response file. It is possible for you to change this file, but you should have no reason to do so, because it is a record of events that have already occurred. Expedite Base/MVS creates a new profile response file when you restart.
- **EDI qualifier and destination tables**. Change only syntax errors in these files. If you change the destination used for a SENDEDI command while the command processes, the resulting information is unreliable.
- **EDI work file (EDIWORK).** Never change this file before restarting. This is a control file used by Expedite Base/MVS to determine which EDI envelopes were sent and which were not.
- **Receive work file (RCVWORK).** Never change this file before restarting. This is a control file used by Expedite Base/MVS to write data received during the processing of a RECEIVE or RECEIVEEDI command.

Resetting the Expedite Base/MVS session

Resetting your Expedite Base/MVS session after an error prevents the program from restarting at the last checkpoint. Under normal conditions, you should never reset your session. If Expedite Base/MVS returns an error other than 28010, resolve the error and restart the program to complete the session.

In the event of an emergency (for example, if the SESSION file becomes damaged, or if you decide you do not want Expedite Base/MVS to complete the requests currently in the message command file), you must reset the session. Follow the steps below:

1. Determine what data has been transmitted.

   Before you can reset your session, you must determine what data Expedite Base/MVS has already sent or received. The best way to do this is to examine your Information Exchange audit trail. An audit trail provides a way of tracking and verifying basic information about the status of messages. You can examine your audit trail using Information Exchange Administration Services, the product used by the Information Exchange Service Administrator to perform administrative tasks.

If you do not want to use the audit trail, you can use the SENT, NOTSENT, and RECEIVED records in the message response file to help determine which data has been sent, not sent, and received. Normally, there is a SENT record for each file or EDI envelope sent and a RECEIVED record for each file or EDI envelope received. Expedite Base/MVS produces a NOTSENT record for every EDI envelope that could not be sent due to a destination verification failure. NOTSENT records are produced only when you use the SENDEDI command with VERIFY(C) or VERIFY(G) specified.

2. Remove the complete requests.

   After you have determined what data has been sent and received and what commands have completed, remove the completed requests from your message command file.

   This can be difficult when you send EDI data, because a single SENDEDI command can send many EDI envelopes from a single file. You have to remove the envelopes that have already been sent from your EDI data file.

   If you do not remove completed SEND requests from your message command file, or if you do not remove the processed envelopes from your EDI file, Expedite Base/MVS sends that data again when you restart it with the RESET command parameter in your EXEC statement. This causes your trading partner to receive two copies of the data. If you do not remove completed receive requests from your message command file, Expedite Base/MVS destroys the data you have already received unless you have used DISP=MOD for the received file.

3. Reset the session.

   After you have removed the completed send and receive requests from your message command file, restart Expedite Base/MVS with the RESET command parameter in your EXEC statement.

Using session-level recovery

Session-level recovery is the default recovery level for Expedite Base/MVS. Session-level recovery does not require additional work files to run, as do the other methods of data recovery. When you use session-level recovery to transmit data and an error occurs, Expedite Base/MVS stops the transmission and produces a SESSIONEND record with a return code. You can check this return code to determine the cause of the error and correct the problem.

Although retransmission of large amounts of data can take several hours, there are advantages to using session-level recovery. When you use session-level recovery, you do not need to be concerned with determining which data sets were successfully sent and which need to be retransmitted. With session-level recovery, if transmission stops because of an error, you send or receive all of the data again.

Session-level recovery considerations

ATTENTION: Do not use the same account and user ID on multiple systems. If you start an Information Exchange session using session-level recovery while another Information Exchange session with the same account and user ID is running, Information Exchange ends the first session and starts the second session. Information Exchange does not deliver data sent in the first session, and does not delete received data from the mailbox. This means that data received in the first session may be received again in error. To prevent this, do not use the same account and user ID on multiple systems.

If you use multiple START and END commands in INMSG, you create an environment similar to that of checkpoint-level recovery. Each END command stops a session. Requests in each session complete even if a subsequent session ends in error. Only sessions that end in error require you to send or receive data again.

There is a limit to the amount of data you can send during a single Information Exchange session. If you plan to send very large amounts of data, check with your representative to find out what the limit is at your Information Exchange installation.

Customers using a leased line may select session-level recovery, because it is very unlikely the line will go down during the data transfer. It is easier to recover if you use session-level recovery, as it is an all-or-nothing data transfer. However, it is recommended that checkpoint-level be used when sending a large number of files, because of the following disadvantage encountered when using session-level.

When using session-level recovery, Information Exchange commits the data sent or received only at the end of the session. When Expedite Base/MVS requests an end to the Information Exchange session, Information Exchange does not respond until the commit process is completed. If a large number of files are transferred during the session, the processing takes longer, especially during prime business hours. It is possible for Expedite Base/MVS to time out while waiting for Information Exchange to complete the process of committing the data for the entire session. In this case, Expedite ends the session with a 29999 return code, “Session end response failure.” It may not be clear, then, whether or not the session ended successfully. One way to determine if the session was successful is to use the CHECK parameter in the START command. For more information, see “START command” on page 134.

If Expedite Base reported the 29999 SESSIONEND return code for a session-level recovery session, you should switch to checkpoint-level, file-level, or user-level recovery for future sessions with a similar number of commands.

You can also use other methods to determine if the previous session was or was not successful.

If you were only receiving files from your Information Exchange mailbox, make sure all data was received by verifying that it is no longer in your mailbox. You can do this by viewing your mailbox with Information Exchange Administration Services or by running a QUERY command to get a list of AVAILABLE response records for each file in your mailbox. If the data is still in your mailbox, switch from session-level recovery to checkpoint-level recovery and run the session again to receive the data.

If you were sending files, you must check the audit trail to see if the files were sent. You can do this by using Information Exchange Administration Services, or by using Expedite to request an audit be sent to your mailbox. If the files were not sent, switch to checkpoint-level recovery and run the session again.

For a large number of files sent or received, session-level is not recommended. Customers have experienced timeout problems when sending or receiving more than 700 files (the size of the files does not matter). Use checkpoint-, user-, or file-level recovery instead.

Processing the message response file for session-level recovery

When you transmit data, Expedite Base/MVS processes your message command file and creates a message response file. The following sections give you the information you need to process the message response file when you use session-level recovery.

Examining response records

Response records are free-format records with the same syntax as commands. See “Understanding command syntax” on page 44 for a description of the command syntax used with Expedite Base/MVS. Response records always start at the beginning of a line. However, parameters in response records can occur in any position in a record and in any order. In addition, all parameters in a response record might not appear.

When examining response records, consider the following:

- Assume a default value if you do not get the response record parameter that you are expecting. This is not an error.
- If a response record parameter has a longer length than you expect, truncate the parameter.
- Be prepared to handle parameters that are split across records. Splitting can occur if the parameter is longer than the record length of your response file.

Checking return codes for session-level recovery

To ensure that Expedite Base/MVS finished processing the message command file, check the return code in the SESSIONEND record.

The following table lists the return codes you can receive, explains them, and tells you what to do when you receive them.

If the SESSIONEND return code is:	Then:	You must do the following:
00000	Expedite Base/MVS finished processing the command file. There were no processing problems, and all of the command RETURN records also contain 0 return codes. All of your requests completed normally.	Nothing.
28010	The Information Exchange session completed normally, but Expedite Base/MVS could not complete all the commands in your message command file.	Check the command RETURN records to see which commands did not complete. Those that completed normally have 0 RETURN records. Those that did not complete have nonzero RETURN records, and, if your system is set up to do so, can display error descriptions.
Anything other than 00000 or 28010	Expedite Base/MVS did not finish processing the command file. Your Information Exchange session failed, and none of the file transfer requests in your message command file completed. No mail has been placed in your trading partner's mailbox or removed from your mailbox.	Correct the problem and restart Expedite Base/MVS. The SESSIONEND record might include an error description to help you find the problem, if your system is set up to display error descriptions. A command RETURN record might contain the same code and description. If OUTMSG contains no return code, check the OUTPRO file for the error.

If the error is due to a problem with a command, such as a syntax error, the command in error is followed by a RETURN record with the same return code as the SESSIONEND record. If you do not find the RETURN record in OUTMSG, the error might be in INPRO, and you might find the RETURN record in OUTPRO.

The SESSIONEND and RETURN records often include an error description. Detailed error descriptions are included in Appendix D, “Expedite Base/MVS messages and codes.”

If the return code is not 00000 or 28010, some data might have been written to your files before the Information Exchange session failed. Check OUTMSG. If the RECEIVE commands you entered have been echoed to OUTMSG, Expedite Base/MVS may have written new data to the files referenced in the RECEIVE commands.

Nonfile transfer requests can complete even if the Expedite Base/MVS SESSIONEND return code is other than 28010 or 00000. These requests include:

- AUDIT
- ARCHIVEMOVE
- CANCEL
- DEFINEALIAS
- GETMEMBER
- LIST
- LISTLIBRARIES
- LISTMEMBERS
- LISTVERIFY
- MESSAGEINFO
- PURGE
- SESSIONINFO
- TESTMSG

If these requests are followed by a RETURN(00000) record in the OUTMSG file, they completed, and you do not need to issue them again. If you issue AUDIT, TESTMSG, or LISTVERIFY again, you receive two identical responses in your mailbox, instead of receiving only one.

Checking the command SENT and RECEIVED records

Expedite Base/MVS produces SENT records for every file sent to Information Exchange. It produces RECEIVED records for every file received from Information Exchange. These records provide you with a history of all Expedite Base/MVS file transfers that took place during the last session.

A single RECEIVE command often produces multiple RECEIVED records, indicating that Expedite Base/MVS has received multiple files as a result of a single command. Also, a single SENDEDI command can produce several SENT records, indicating that several EDI envelopes were sent with a single command.

If there is no SENT record, no file was sent. If there is no RECEIVED record, no file was received.

OUTMSG examples

The following sections contain examples to help you understand OUTMSG processing. The INMSG file used as a basis for the OUTMSG examples follows.

NOTE: In the SESSIONEND(28010) example and the SESSIONEND (not 00000 or 28010) example, INMSG has been altered to include a syntax error that provides you with several different OUTMSG files. Each OUTMSG has a different SESSIONEND record.

INMSG

```
#Send a file named user01.test.data to act1 user01 in user class tdata
```

```
send fileid(user01.test.data) account(act1) userid(user01)
class(tdata);
#Send an EDI file called user01.edidata.file in user class editest
sendedi fileid(user01.edidata.file) class(editest);
#Receive an EDI file with user class #e2 into user01.ediout.file
receiveedi fileid(user01.ediout.file) class(#e2);
#Receive a file called user01.test.out with user class tdata
receive fileid(user01.test.out) class(tdata);
```

Example of normal completion SESSIONEND(00000)

In the following OUTMSG example, all commands from the INMSG shown above complete successfully:

```
AUTOSTART SESSIONKEY(HSXVNW1M);
STARTED LASTSESS(0) RESPCODE(00000) SESSIONKEY(HSXVNW1M)
IEVERSION(04) IERELEASE(07);
RETURN(00000);

#Send a file named user01.test.data to act1 user01 in user class tdata#

send fileid(user01.test.data) account(act1) userid(user01)
class(tdata);
SENT UNIQUEID(HSXVN3A5) LENGTH(80);
RETURN(00000);

#Send an EDI file called user01.edidata.file in user class editest#

sendedi fileid(user01.edidata.file) class(editest);
SENT UNIQUEID(HSXVOACZ) LENGTH(2260) ACCOUNT(ACT1) USERID(USER01)
EDITYPE(X12)
DESTINATION(DUNS01) QUALIFIER(01) CONTROLNUM(0000022223) CLASS(EDITEST)
MSGNAME(00022223) MSGSEQNO(00001);
RETURN(00000);

#Receive an EDI File with user class #e2 into user01.ediout.file#

receiveedi fileid(user01.ediout.file) class(#e2);
RECEIVED ACCOUNT(ACT1) USERID(USER01) CLASS(#E2) CHARGE(5)
RECEIVER(ACT1 DEPT01) RECVQUAL(01) SENDER(ACT2 DEPT02)
SENDQUAL(01) CONTROLNUM(000022228) FILEID(USER01.EDIOUT.FILE)
MSGDATE(980326) MSGDATELONG(19980326) MSGTIME(182209)
MSGSEQO(000021) MSGNAME(00022223) MSGSEQNO(00001)
SESSIONKEY(HSXVNW1M) DELIMITED(E) TIMEZONE(L)
SYSNAME(EXMV000T) SYSLEVEL(450) DATATYPE(E) EDITYPE(X12)
SENDERFILE(USER01.EDIDATA.FILE) SENDERLOC(3390 /TSO203)
FILEDATE(980326) FILEDATELONG(19980326) RECFM(F) RECLEN(80) RECDLM(E)
UNIQUEID(75150113) SYSTYPE(21) SYSVER(4);
RETURN(00000);

#Receive a file called user01.test.out with user class tdata#

receive fileid(user01.test.out) class(tdata);
RECEIVED ACCOUNT(ACT1) USERID(USER01) CLASS(TDATA) CHARGE(5)
FILEID(USER01.TEST.OUT) MSGDATE(980326) MSGDATELONG(19980326)
MSGTIME(182208) MSGSEQO(000023)SESSIONKEY(HSXVNW1M) DELIMITED(N)
SYSNAME(EXMV000S) SYSLEV(450) TIMEZONE(L) DATATYPE(E)
EDITYPE(UNFORMATTED) SENDERFILE(USER01.TEST.DATA)
SENDERLOC(3390 /TSO203) FILEDATE(980326) FILEDATELONG(19980326)
RECFM(FB) RECLEN(80) RECDLM(N) UNIQUEID(57586838) SYSTYPE(21)
SYSVER(4);
```

```
RETURN(00000);

AUTOEND;
RETURN(00000);
SESSIONEND(00000);
```

Example of a minor error SESSIONEND(28010)

In the following OUTMSG example, you see the result of a typographical error. The programmer intends to enter **user01.edidata.file** as part of the SENDEDI command, but types **user01.edidata.tile** instead. The result is a 23410 error after the SENDEDI command and a 28010 error in the SESSIONEND record. Note that the other requests in INMSG process successfully.

```
AUTOSTART SESSIONKEY(HXTB08G);
STARTED LASTSESS(0) RESPCODE(00000) SESSIONKEY(HXTB008G)
IEVERSION(04) IERELEASE(07);
RETURN(00000);

#Send a file named user01.test.data to act1 user01 in user class tdata#
send fileid(user01.test.data) account(act1) userid(user01)
class(tdata);
SENT UNIQUEID(HXTB0157) LENGTH(80);
RETURN(00000);

#Send an EDI file called user01.edidata.file in user class editest#

sendedi fileid(user01.edidata.tile) class(editest);

RETURN(23410) ERRDESC(EDI send file not found.)
REASON(1044  Dataset USER01.EDIDATA.TILE not found.)
ERRTEXT(EXPLANATION: Expedite Base could not open the file indicated in
the)
ERRTEXT(SENDEDI command. There might be a REASON parameter that gives
more)
ERRTEXT(information about why the file could not be opened.)
ERRTEXT(USER RESPONSE: If there is a REASON parameter, check it for
more)
ERRTEXT(information about the error. Also, make sure the file
specified)
ERRTEXT(is not empty. Check the FILEID parameter and JCL for)
ERRTEXT(errors and try the command again.);

#Receive an EDI file with user class #e2 into user01.ediout.file#

receiveedi fileid(user01.ediout.file) class(#e2);
RECEIVED ACCOUNT(ACT1) USERID(USER01) CLASS(#E2) CHARGE(5)
RECEIVER(ACT1 DEPT01) RECVQUAL(01)
SENDER(ACT2 DEPT02) SENDQUAL(01) CONTROLNUM(000022228)
FILEID(USER01.EDIOUT.FILE) MSGDATE(980326) MSGDATELONG(19980326)
MSGTIME(182446) MSGSEQO(000025) MSGNAME(00022223) MSGSEQNO(00001)
SESSIONKEY(HXTBO08G) DELIMITED(E) TIMEZONE(L) SYSNAME(EXMV000T)
SYSLEVEL(450) DATATYPE(E) EDITYPE(X12) SENDERFILE(USER01.EDIDATA.FILE)
SENDERLOC(3390 /TSO203) FILEDATE(980326) FILEDATELONG(19980326)
RECFM(F) RECLEN(80) RECDLM(E) UNIQUEID(75150113) SYSTYPE(21) SYSVER(4);
RETURN(00000);

#Receive a file called user01.test.out with user class tdata#

receive fileid(user01.test.out) class(tdata);
RECEIVED ACCOUNT(ACT1) USERID(USER01) CLASS(TDATA) CHARGE(5)
FILEID(USER01.TEST.OUT) MSGDATE(980326) MSGDATELONG(19980326)
```

```
MSGTIME(182445) MSGSEQO(000027) SESSIONKEY(HXTBO08G) DELIMITED(N)
SYSNAME(EXMV000S) SYSLEVEL(450) TIMEZONE(L) DATATYPE(E)
EDITYPE(UNFORMATTED) SENDERFILE(USER01.TEST.DATA)
SENDERLOC(3390 /TSO203) FILEDATE(980326) FILEDATELONG(19980326)
RECFM(FB) RECLEN(80) RECDLM(N) UNIQUEID(57586838) SYSTYPE(21)
SYSVER(4);
RETURN(00000);

AUTOEND;
RETURN(00000);
SESSIONEND(28010)
ERRDESC(The IE session completed normally but not all requests were
processed.)
ERRTEXT(EXPLANATION: The Information Exchange session completed
normally, but)
ERRTEXT(not all of the requests in the message command file, INMSG,
processed)
ERRTEXT(or some requests generated warnings.)
ERRTEXT(USER RESPONSE: Check the message response file, OUTMSG, to see)
ERRTEXT(which requests did not process normally. Correct those
requests)
ERRTEXT(in a new INMSG file, and retry the program.);
```

Example of a severe error SESSIONEND (not 00000 or 28010)

In the following OUTMSG example, you see the result of a typographical error. The programmer intends to enter the SENDEDI command, but types **sendedk** instead. The result is a 15040 error after **sendedk** and a 15040 error in the SESSIONEND record. Note that although the SEND command preceding **sendedk** completed, no data is placed in the recipient's Information Exchange mailbox, because the session does not complete.

```
AUTOSTART SESSIONKEY(ISS5HK8S);
STARTED LASTSESS(0) RESPCODE(00000) SESSIONKEY(ISS5HK8S)
IEVERSION(04) IERELEASE(07);
RETURN(00000);

#Send a file named user01.test.data to act1 user01 in user class tdata

send fileid(user01.test.data) account(act1) userid(user01)
class(tdata);
SENT UNIQUEID(ISS5HL10) LENGTH(80);
RETURN(00000);

#Send an EDI file called user01.edidata.file in user class editest

sendedk
RETURN(15040) ERRDESC(Command not recognized.)
ERRTEXT(EXPLANATION: You specified an unrecognized command in the
command file.)
ERRTEXT(USER RESPONSE: Check the message response file, OUTMSG,)
ERRTEXT(profile response file, OUTPRO, or)
ERRTEXT(response work file, OUTWORK, to determine which command produced
the error.)
ERRTEXT(Correct the appropriate command file and retry the program.);
SESSIONEND(15040) ERRDESC(Command not recognized.)
ERRTEXT(EXPLANATION: You specified an unrecognized command in the
command file.)
ERRTEXT(USER RESPONSE: Check the message response file, OUTMSG, profile
response)
```

```
ERRTEXT(file, OUTPRO, or response work file, OUTWORK, to determine which)
ERRTEXT(command produced the error. Correct the appropriate command)
ERRTEXT(file and retry the program.);
```

Chapter 7

Expedite Base/MVS commands

To use Expedite Base/MVS, you need to understand the basic command syntax and how to use the profile and message commands. The following sections describe the API command syntax and provide you with details on the use of each command.

Command syntax

An example of Expedite Base/MVS command syntax is: #Comment or description

```
command parameter(value) parameter(value) ...parameter(value);
```

Descriptions of the syntax elements are as follows:

Element	Description
#	Defines or delimits a comment line. You can type any information you like after a #, and Expedite Base/MVS ignores the characters that follow it on the same line. If you include a # in a parameter value, Expedite Base/MVS knows from the value's context that the # is part of a command and does not ignore the parameter value or the characters that follow it.
command	Identifies the Expedite Base/MVS command.
parameter	Identifies a parameter of the associated command.
value	Defines the value associated with the parameter.
...	In the previous example, the ellipsis (...) indicates that you can specify as many parameters as necessary. (It is not part of the syntax.)
;	Ends the command.

You can type Expedite Base/MVS commands and parameters in uppercase or lowercase letters. The commands and parameters can span several lines in a command file. However, the following limitations apply:

- You must type the entire command name, for example, IDENTIFY, on a single input line.
- You must type the entire parameter name, for example, IEACCOUNT, on a single input line.
- A left parenthesis must immediately follow each parameter. Do not use spaces between parameter names and values. For example, type **inaccount(acct)** rather than **inaccount (acct)**.
- End each command with a semicolon.

Command syntax examples

Each section describing a command in this chapter contains an example of that command's syntax. The examples are formatted as follows:

- All parameter values are shown in italics.
- Required parameter values are shown in boldface.
- Default parameter values are underlined.
- Some parameter values include *blank* as an option.

  The word *blank* in a syntax example or description indicates a space containing no characters. It does not mean that you should enter the word blank.
- In some examples, you have a choice between parameters or groups of parameters. Your choices, in cases like these, are separated from the rest of the example with extra blank lines and are separated from each other with the word or.

  Choose one line of parameters from any group of lines separated in this way. If there are required parameters in more than one of the choices, you still need to choose only one line.

Profile commands

The profile commands give you a simple method for creating and maintaining your profile. You place these commands in the INPRO command file. Expedite Base/MVS places return codes reflecting the completion of these commands in the OUTPRO response file.

The profile commands are listed here with the page numbers on which detailed descriptions of them appear:

- IDENTIFY, page 68
  Gives you a way to set up the fields in the profile related to your account, user ID, and password information. With this command, you can also specify the time zone.
- SNACOMM, page 71
  If you choose to use SNA communications, specifies the logical unit (LU) names for your own system and for the Information Exchange Common Front End to which you connect.
- SSL, page 72
  Indicates whether you intend to use SSL communications.

- TCPCOMM, page 73
  If you choose to use TCP/IP communications, specifies the host and port for the Information Exchange Common Front End to which you connect.
- TRACE, page 74
  Specifies the information Expedite Base/MVS should record in the trace file during the session. Possible traces might include LINK, PROTOCOL, or BASE.
- TRANSMIT, page 75
  Specifies transmission control parameters.

Detailed information on each of the commands listed previously is provided on the following pages.

IDENTIFY command

The IDENTIFY command sets up the Information Exchange account, user ID, and password information, as well as the time zone. This command is required when you specify AUTOSTART(y) in the TRANSMIT command. For more information, see “Required IDENTIFY parameters” on page 13.

Syntax

```
identify

ieaccount(IE account) ieuserid(IE user ID)
iepassword(IE password) niepassword(new IE password)
timezone(time zone) codepage(code page)

keyringfile(fspec)
   OR
keyringfile(fspec) keyringpassword(password)
   OR
keyringfile(fspec) keyringstashfile(fspec);
```

Parameters

ieaccount
Information Exchange account name. Use 1 to 8 alphanumeric characters.

ieuserid
Information Exchange user ID. Use 1 to 8 alphanumeric characters.

iepassword
Information Exchange password. Use 1 to 8 alphanumeric characters.

niepassword
New Information Exchange password. If you specify this value, the Information Exchange password changes upon completion of the Information Exchange session. If the Information Exchange session ends in an error, the password does not change. Use 1 to 8 alphanumeric characters.

NOTE: Remove this parameter and change the IEPASSWORD parameter after you change your password.

timezone
Your local time zone. You can enter one of the time zone codes listed here, or you can specify an offset from Greenwich mean time (GMT) by indicating the number of hours and minutes east or west of the Greenwich meridian. The format for specifying the hours and minutes is *ehhmm* or *whhmm*, where:

Type:	To Indicate:
e	east
w	west
hh	hours
mm	minutes

For example, to specify eastern daylight time, you can enter either edt or w0400, where w indicates west, and 0400 indicates 4 hours and 0 minutes.

Type:	To indicate:
ahd	W1000 (Hawaii standard time)
ahs	W1000 (Hawaii standard time)
ast	W0400 (Atlantic standard time)
bst	E0100 (British summer time)
cdt	W0500 (Central daylight time)
cst	W0600 (Central standard time)
ead	E1000 (Eastern Australia daylight time)
edt	W0400 (Eastern daylight time)
emt	E0200 (Eastern Mediterranean time)
est	W0500 (Eastern standard time)
gmt	E0000 (Greenwich mean time)
jst	E0900 (Japanese standard time)
mdt	W0600 (Mountain daylight time)
mst	W0700 (Mountain standard time)
pdt	W0700 (Pacific daylight time)
pst	W0800 (Pacific standard time)
wed	E0200 (Western Europe daylight time)
wes	E0100 (Western Europe standard time)
ydt	W0800 (Alaska daylight time)
yst	W0900 (Alaska standard time)

Use 1 to 5 alphanumeric characters. The default is **gmt.**

codepage
The code page you want to associate with the data you send. If you specify the code page, Expedite Base/MVS includes the value in the CDH when you send data. Recipients can decide whether they need to provide special data translation facilities. Use 1 to 3 numeric characters.

keyringfile
The name of the key database HFS file or the RACF managed key ring. This parameter is required if ENABLESSL= y on the SSL command, and must be created prior to invoking Expedite Base/MVS. Use 2 to 251 characters.

If you specify a key database HFS file, you must provide either the password or the stash file that contains the password. For more information on creating a key database, see Chapter 11, “Communicating with Information Exchange using SSL.” You must fully qualify this name including the file extension. A fully-qualified key database HFS file name must be between 2 and

251 characters, and should either have no extension or an extension of **.kdb**. The maximum allowed length of a database name is 247characters excluding any extension characters. The key database name may not end with **.rdb** or **.sth**.

If you specify a RACF managed key ring, no other parameters are necessary as access authority is controlled through RACF. The format for specifying a RACF managed key ring is user ID/keyring.

The values in this parameter are case sensitive.

keyringpassword
The password for the key database. This value is established when the key database is created. Use 1 to 128 characters. If you use this parameter, do not use keyringstashfile.

keyringstashfile
The name of the key database password stash file. The name of this file must include a **.sth** extension. You must create this file prior to invoking Expedite Base/MVS.

IDENTIFY command example

```
identify ieaccount(account) ieuserid(userid) iepassword(mypswd)
timezone(est)
keyringfile(/u/userid/keyring.kdb);
```

SNACOMM command

Expedite Base/MVS uses the SNACOMM command to indicate the names for your LUs and the Information Exchange Common Front End to which you connect. Your system programmer can provide you with these names. This command is required for using SNA communications. For more information, see “Required SNACOMM parameter” on page 14.

Syntax

```
snacomm

ieluname(IE LU name) userluname(user LU name) ielumode(IE LU mode)
rusize(RU size);
```

Parameters

ieluname
LU name for the Information Exchange Common Front End. Use 1 to 8 alphanumeric characters. The default is **ibm0rely**, which is the Information Exchange Common Front End LU name in the United States.

NOTE: If you are in a country other than the United States, contact your marketing representative for more information.

userluname
LU name for your application. It must be defined to VTAM® on the MVS system. If more than one copy of Expedite Base/MVS is executed concurrently, you can repeat this parameter to define a group of LU names. Expedite Base/MVS then selects the first unused LU name from the group. You can specify this parameter as many as 99 times. If you specify the SNACOMM command more than once, the USERLUNAMEs on the last SNACOMM command override any previous USERLUNAMEs specified. Use 1 to 8 alphanumeric characters. This parameter is required.

ielumode
Mode name used to communicate with the Information Exchange Common Front End. Use 1 to 8 alphanumeric characters. The default is **iinappc**.

rusize
Size of SNA request units. Valid values are from 256 to 3840. The default value is **3840**. Let this parameter default unless your system programmer tells you otherwise.

SNACOMM command example

```
snacomm userluname(explunm1) userluname(explunm2);
```

SSL command

Use the SSL command to enable secure socket layer (SSL) communication. COMMTYPE(t) must be specified on the TRANSMIT command to enable SSL communications.

Syntax

```
ssl
enablessl(y|n);
```

Parameters

enablessl
Indicates whether SSL should be enabled for sending and receiving files.

y Expedite Base/MVS will attempt to establish a connection with Information Exchange using SSL.

NOTE: Make sure that the TCP/IP address of the Information Exchange Secure Server is included in the TCPCOMM command.

n Expedite Base/MVS will not attempt to establish a connection with Information Exchange using SSL. This is the default.

SSL command example

```
ssl enablessl(y);
```

TCPCOMM command

Use the TCPCOMM command to specify the TCP/IP host address and port to use when connecting to a network. You can enter up to five sets of host/port combinations. You correlate the members of a set by specifying a number from 1 to 5 at the end of the parameter. This command is required to use TCP/IP communications, including SSL sessions.

The parameter values specified on this command are only used when the connection is established. Once the session has been established, Expedite Base/MVS will not try connecting with another host/port combination if the session fails.

Syntax

`tcpcomm`

`ietcphost`*`n`*`(`*`IE TCP/IP hostN`*`)` **`ietcpportn`**`(`*`IE TCP/IP port IDN`*`);`

Parameters

ietcphostn
Indicates a host IP address that Expedite Base/MVS can try when connecting to Information Exchange. To specify an IP address, use the format *nnn.nnn.nnn.nnn*, where each *nnn* represents a numeric value between 0 and 255. This is a required parameter.

The valid values for n are **1** to **5**.

ietcpportn
Indicates the port ID Expedite Base/MVS uses when connecting to Information Exchange. Use up to five numeric digits to specify a number from 1 to 65535. This is a required parameter.

The valid values for n are **1** to **5**.

TCPCOMM command example

```
tcpcomm ietcphost1(32.76.15.6) ietcpport1(3001);
```

Results: Expedite Base/MVS will attempt to connect to host 32.76.15.6 port 3001.

NOTE: While the TCPCOMM command allows you to specify more than one host/port combination, your Information Exchange installation may only have one valid host/port combination defined to which you can connect. Check with Customer Care for a list of valid combinations.

The TCP/IP address for SSL connections is different from the TCP/IP address for non-SSL connections.

TRACE command

The TRACE command specifies what information the trace file records during a session. Possible traces are LINK, PROTOCOL, IOFILE, and BASE. All traces are used by GXS for problem determination.

If you request PROTOCOL, BASE, or IOFILE traces, you must create a file to hold the trace information and refer to it with the ddname BASETRC. If you request a link trace, you must create a file to hold the link trace information and refer to it with the ddname LINKTRC.

Syntax

`trace`

```
protocol(n|y) link(n|y) base(n|y) iofile(n|y);
```

Parameters

protocol
Indicates whether the trace file should contain Information Exchange protocol information.

n	Do not include the Information Exchange protocol information in the trace file. This is the default.
y	Include the Information Exchange protocol information in the trace file.

link
Indicates whether or not Expedite Base/MVS should trace link protocol. Expedite Base/MVS places the link protocol data in a separate file referred to with the ddname LINKTRC.

n	Do not include the link information in the trace file. This is the default.
y	Include the link information in the trace file.

base
Indicates whether the trace file should contain the Expedite Base/MVS module information.

n	Do not include the Expedite Base/MVS module information in the trace file. This is the default.
y	Include the Expedite Base/MVS module information in the trace file.

iofile
Indicates whether the trace file should show the parsing of the INPRO and INMSG files. This trace can be useful for finding the cause of syntax errors.

n	Do not include the command parsing information in the trace file. This is the default.
y	Include the command parsing information in the trace file.

TRACE command example

```
trace protocol(y);
```

TRANSMIT command

You can use the TRANSMIT command to specify a level of data recovery. You can also use this command to specify whether to start and end an Information Exchange session automatically. For information on session-level recovery, see Chapter 6, "Using session-level recovery." For information on alternative data recovery methods, see Chapter 5, "Using checkpoint-level, file-level, and user-initiated recovery."

Syntax

```
transmit

autostart(y|n) autoend(y|n)
msgsize(message size) commitdata(commit data)
maxmsgs(maximum messages) recovery(s|c|f|u) timeout(timeout)
commtype(s|t);
```

Parameters

autostart
Indicates whether Expedite Base/MVS should start an Information Exchange session automatically, using the account ID and user ID on the IDENTIFY command. Expedite Base/MVS automatically starts a session only when it begins to process INMSG. If you want to start multiple Information Exchange sessions in INMSG, you must specify both AUTOSTART(n) and AUTOEND(n).

n	Expedite Base/MVS does not start the Information Exchange session automatically. You must include the START command in the message command file (INMSG).
y	Expedite Base/MVS sends the Information Exchange session start command automatically. Do not include the START command in the message command file (INMSG) when you transmit data. This is the default.

autoend
Indicates whether Expedite Base/MVS sends an Information Exchange session end command when it finishes processing your command file. Expedite Base/MVS automatically ends a session only when it finishes processing INMSG. If you want to start multiple Information Exchange sessions in INMSG, you must specify both AUTOSTART(n) and AUTOEND(n).

n	Expedite Base/MVS does not automatically end the Information Exchange session. You must include the END command in the INMSG file.
y	Expedite Base/MVS automatically ends the Information Exchange session. Do not include the END command in INMSG. This is the default.

msgsize
Segments the data for sending. Your trading partner can take checkpoints only for the message size you specify with this parameter. If you use a large value, your trading partner cannot take frequent checkpoints.

Valid values are 1000 through 99999. The default value is **47000** bytes. Lower values permit the receiving interface to complete more frequent checkpoints while receiving the data.

NOTE: You should use the default value when communicating with a slower receiver, such as a PC.

commitdata
This parameter applies only to checkpoint-level recovery. It is ignored for session-level, file-level, or user-initiated recovery.

This parameter indicates the maximum amount of data Expedite Base/MVS sends between checkpoints. For maximum efficiency, the value for the COMMITDATA parameter should be an even multiple of the MSGSIZE value. Lower values result in less retransmission of data if there is a communication failure. Higher values provide faster data transmission.

Valid values are 1000 through 9999999 and must be at least as large as MSGSIZE. The default value is **141000**.

maxmsgs
Maximum number of message segments Expedite Base/MVS can receive between Information Exchange commits. The valid values are 1 to 1000. The larger the number specified, the more data Information Exchange sends to you without committing it. A lower number causes more frequent data commits. The default value is **10**. Unless you experience frequent losses of connection, do not change this parameter.

recovery
Indicates what type of data recovery will be used.

s	Session-level recovery. This is the default. For more information, see Chapter 6, “Using session-level recovery.”
c	Checkpoint-level recovery. For more information, see Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery.”
f	File-level recovery. For more information, see Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery.”
u	User-initiated recovery. For more information, see “COMMIT command” on page 85 and Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery.”

timeout
Number of minutes Expedite Base/MVS waits for a response from Information Exchange before generating an error. Valid values are 0 through 99. The default value is **2**.

NOTE: If you specify 0, the maximum value is used.

commtype
The COMMTYPE parameter allows you to specify the type of communication being used.

s	Use SNA LU 6.2 to communicate with Information Exchange. This is the default.
t	Use TCP/IP to communicate with Information Exchange. This value is also used for SSL communications.

TRANSMIT command example

```
transmit recovery(c);
```

Message commands

Use the message commands to send and receive files, control your Information Exchange mailbox, and define lists of users that you can use when sending and receiving files. Enter message commands in the Expedite Base/MVS message command file, INMSG. The message commands are listed here with the page numbers on which their detailed descriptions appear:

- ARCHIVEMOVE, page 79
  Moves messages from the Information Exchange short-term archive to your mailbox. Expedite Base/MVS places a MOVED record in the response file as a result of the ARCHIVEMOVE command.
- AUDIT, page 80
  Loads an audit trail in your mailbox.
- CANCEL, page 83
  Cancels previously sent files if the receiver has not retrieved them from the mailbox.
- COMMIT, page 85
  Sends a commit to Information Exchange and processes the response.
- DEFINEALIAS, page 86
  Defines a new alias, redefines an existing alias, or changes or deletes an existing alias table.
- END, page 89
  Ends an Information Exchange session.
- GETMEMBER, page 90
  Copies a library member from an existing Information Exchange library to an Information Exchange mailbox.
- LIST, page 93
  Sets up a list of account and user IDs that you can use to send and receive files.
- LISTLIBRARIES, page 95
  Returns a list of Information Exchange libraries to which you have access.
- LISTMEMBERS, page 96
  Returns a list of members within an Information Exchange library.
- LISTVERIFY, page 97
  Gives you information about lists.
- MSGINFO, page 100
  Gives you information about messages currently available in your mailbox.
- PURGE, page 101
  Deletes a specific file from your mailbox.
- PUTMEMBER, page 102
  Adds a member to an existing Information Exchange library.
- QUERY, page 105
  Gets information about all the messages in your mailbox. You can also use QUERY to see the CDH information associated with each message, if desired.
- RECEIVE, page 106
  Receives all files or specific files, including free-format messages.

- RECEIVEEDI, page 111
  Receives EDI-formatted files. Expedite Base/MVS supports X12, UCS, EDIFACT, and UN/TDI standards.
- RECEIVESTREAM, page 115
  Receives data directly into the message response file (OUTMSG). When Expedite Base/MVS receives data using the RECEIVESTREAM command, it places the data immediately after the RECEIVED record in the OUTMSG file.
- SEND, page 118
  Sends files, including free-format messages that you saved into a file, to a user or a list of users.
- SENDEDI, page 124
  Sends EDI-formatted files. Expedite Base/MVS supports X12, UCS, EDIFACT, and UN/TDI standards.
- SENDSTREAM, page 129
  Sends data from the message command file (INMSG). Place the data immediately after the SENDSTREAM command in the command file.
- SESSIONINFO, page 133
  Gives you information about your Information Exchange session.
- START, page 134
  Starts an Information Exchange session.
- TESTMSG, page 136
  Tells Information Exchange to load one or more test messages into the mailbox.

Detailed information on each of the commands listed previously is provided on the pages that follow.

ARCHIVEMOVE command

Use the ARCHIVEMOVE command to move messages from the Information Exchange short-term archive to your mailbox. Expedite Base/MVS places a MOVED record in the response file as a result of the ARCHIVEMOVE command.

Syntax

`archivemove`

`archiveid``(`*`archive ID`*`);`

Parameter

archiveid
This is the archive reference identifier for the messages you want to move from the archive to your mailbox. The archive reference identifier is the value you originally specified in the ARCHIVEID parameter of the RECEIVE, RECEIVEEDI, or RECEIVESTREAM command when the message was received. If you specified no ARCHIVEID when the message was received, the archive reference identifier which Information Exchange assigned was indicated in the SESSIONKEY parameter of the RECEIVED record. Use 1 to 8 alphanumeric characters. This parameter is required.

ARCHIVEMOVE command example

```
archivemove archiveid(myrefid);
```

AUDIT command

Information Exchange provides an audit trail that you can retrieve. Use the AUDIT command to tell Information Exchange to load audit information into your mailbox. When an audit report becomes available, you must issue a RECEIVE command to retrieve it. The audit information comes from the *SYSTEM* account and the *AUDITS* user ID in message user class #SAUDIT.

For details on audit information, see the Information Exchange Interface Programming Guide. A sample audit format program is included on the Expedite Base/MVS program tape. A sample audit report, which the audit program generates, is included in Appendix C, "Sample audit report."

NOTE: Information Exchange does not make audits available immediately. Therefore, you cannot successfully issue the RECEIVE command immediately after giving the AUDIT command. Audits become available during a subsequent session.

Syntax

```
audit

account(account) userid(user ID)
   or
sysid(system ID) account(account) userid(user ID)
   or
alias(alias) aliasname(alias name)

altuserid(alternate user ID|?)
msgtype(s|r|b) class(class)
startdate(starting date) enddate(ending date) status(blank|u|p|d)
timezone(l|g) level(1|2|3)
altacct(account);
```

Parameters

account
Account name of an Information Exchange trading partner. Expedite Base/MVS uses this field together with the USERID parameter to indicate that you want only audit records for message exchanges with this account and user ID. If you specify an ACCOUNT parameter, you must also specify a USERID parameter. Use 1 to 8 alphanumeric characters.

userid
User ID of an Information Exchange trading partner. Expedite Base/MVS uses this parameter together with the ACCOUNT parameter to indicate that you want only audit records for message exchanges with this user ID and account. If you specify a USERID parameter, you must also specify an ACCOUNT parameter. Use 1 to 8 alphanumeric characters.

sysid
System ID of an Information Exchange trading partner. You need the system ID only if you specify the ACCOUNT and USERID parameters for a trading partner on another Information Exchange system. If you specify a SYSID parameter, you must also specify the ACCOUNT and USERID parameters. Use 1 to 3 alphanumeric characters.

alias
Table type and table name of an alias table. Use this parameter with the alias name to indicate that you want only audit records for messages exchanged with a single trading partner.

blank	You did not use an alias name. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

NOTE: You create and maintain alias tables by using Information Exchange Administration Services (see Using Information Exchange Administration Services), or by using the DEFINEALIAS command (see "DEFINEALIAS command" on page 86).

aliasname
Alias name defined in the alias table. Use this parameter to indicate that you want audit records only for messages exchanged with a particular trading partner. Use 1 to 16 alphanumeric characters.

altuserid
Indicator that specifies you want to retrieve audit records for other users in your account. To retrieve audit records for another user in your account, put the user's ID here. To retrieve audit records for all user IDs in your account, use ? as the parameter value.

If you want to retrieve audit records for several specific users, you must specify each user ID in a separate AUDIT command.

NOTE: Information Exchange puts an error message in your mailbox and does not use this field if you do not have service administrator authorization.

msgtype
Indicates the type of audit records to retrieve.

s	Retrieve all audit records for files you sent.
r	Retrieve all audit records for files sent to you.
b	Retrieve all audit records for files you have sent, and those that have been sent to you. This is the default.

class
Indicator that specifies that you want to retrieve only those audit records for files with the user classification given here. Use 1 to 8 alphanumeric characters.

startdate
Starting date of a time range for the messages and files for which you want audit information. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 000102 (January 2, 1900).

enddate
Ending date of a time range for the messages and files for which you want audit information. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 420916 (September 16, 2042).

status
Indicator that specifies that you want only audit records for messages with a particular status.

blank	Retrieve audit records for all files. This is the default.
u	Retrieve audit records only for files that have not been delivered.
p	Retrieve audit records for purged files only.
d	Retrieve audit records for delivered files only.

timezone
Date and time point of reference used by the STARTDATE and ENDDATE parameters.

g	Greenwich mean time.
l	Your local time, as specified by the TIMEZONE parameter of the IDENTIFY command. This is the default.

level
Indicator that specifies the information included in each audit record.

1	Retrieve basic audit information. This is the default.
2	Retrieve basic audit information plus enhanced time fields.
3	Retrieve level 2 audit information, plus the EDI control number, for one or more users in another account.

NOTE: If you specify an audit level that is not supported, then Information Exchange uses the default, level 1, and puts a level 1 audit in your mailbox in response to your request.

For more information, see the Information Exchange Interface Programming Guide.

altacct
Indicates the alternate account for which you want to receive an audit. ALTACCT can be one to eight characters. If you specify ALTACCT, you must specify ALTUSERID.

This field allows you to request audit records for an account ID or user ID other than your own. If left blank, audit records for your own account will be returned. If a value is specified in this parameter, ALTUSERID also must be specified. You must have authority to view audit records for the alternate account ID or user ID. Note that this field is only valid for level 3 audit requests.

AUDIT command example

```
audit account(acct) userid(user01) class(#e2) level(2);
```

Results: This command places a level 2 audit report in your mailbox. This audit will include information about all files with a class of *#e2* sent to and received from account *acct* and user ID *user01*, or any files that user ID might have purged.

CANCEL command

Use the CANCEL command to cancel files that you previously sent by Information Exchange to a single account and user ID, a list of users, or a destination specified by an alias name. You can cancel these files only if the receiver has not retrieved them from Information Exchange. Only the sender of a file can request to cancel the file from the receiver's mailbox.

Syntax

```
cancel

alias(alias) aliasname(alias name)
   or
account(account) userid(user ID)
   or
listname(list name)

priority(blank|p) msgname(message name)
msgseqno(message sequence number) class(class)
timezone(l|g) ack(blank|h|t)
startdate(starting date) starttime(starting time)
enddate(ending date) endtime(ending time);
```

Parameters

alias
Alias table type and table name.

blank	No alias table is being used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

NOTE: You create and maintain alias tables by using Information Exchange Administration Services (see Using Information Exchange Administration Services), or by using the DEFINEALIAS command (see "DEFINEALIAS command" on page 86).

aliasname
Alias name defined in the alias table. Use 1 to 16 alphanumeric characters.

account
Account name of a single-destination user ID. Use 1 to 8 alphanumeric characters.

userid
Destination user ID. Use 1 to 8 alphanumeric characters.

listname
Name of a previously defined list of accounts and user IDs. Use 1 to 8 alphanumeric characters.

priority
Option you can use to limit the files canceled to those with the priority specified here.

blank	Normal priority. This is the default.
p	High priority.

msgname
Option you can use to limit the files canceled to those with the message name specified in this parameter. Use 1 to 8 alphanumeric characters.

msgseqno
Option you can use to limit the files canceled to those with the message sequence number specified in this parameter. Use 1 to 5 alphanumeric characters.

class
Option you can use to limit the messages canceled to those with the user classification specified in this parameter. Use 1 to 8 alphanumeric characters.

timezone
Date and time reference in the STARTDATE, STARTTIME, ENDDATE, and ENDTIME parameters. Enter **g** for Greenwich mean time. Your local time as specified on the TIMEZONE parameter of the IDENTIFY command. This is the default.

ack
Types of acknowledgment messages you want to receive regarding the cancellation.

blank	Information Exchange creates no acknowledgments regarding the cancellation of a message group. This is the default.
h	Acknowledgments include only header information.
t	Acknowledgments include both header and text information.

startdate
Starting date of a time range for the files sent to Information Exchange that you want to cancel. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 000102 (January 2, 1900).

starttime
Start time of a time range for the files sent to Information Exchange that you want to cancel. For a file to qualify for cancellation, the submit time of the beginning of the file must fall within this time range. Use the *HHMMSS* format for the start time. The default value is determined by Information Exchange, and is currently 000000 (0 hours).

enddate
Ending date of a time range for the files sent to Information Exchange that you want to cancel. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 420916 (September 16, 2042).

endtime
Ending time of a time range for files sent to Information Exchange that you want to cancel. For a file to qualify for cancellation, the submit time of the beginning of the file must fall within this time range. Use the *HHMMSS* format for the end time. The default value is determined by Information Exchange, and is currently 235347 (23:53:47).

CANCEL command example

```
cancel account(acct) userid(user01) class(special);
```

Results: This command removes any files in the mailbox for account *acct* and user ID *user01* that you sent with a class of special and that have not yet been received.

COMMIT command

Use the COMMIT command to send a commit to Information Exchange and process the response. In order for Expedite Base/MVS to perform the commit, a SEND, SENDEDI, or PUTMEMBER command must be requested between COMMIT commands or between a session start and a COMMIT command.

The message response record associated with the COMMIT command is the RETURN record.

Checkpoints are taken automatically:

- After each COMMIT command, unless there is nothing to commit
- At the end of each session, even if you have not specified a COMMIT command

  NOTE: In user-initiated recovery, if the data set associated with the INMSG ddname is a SYSIN data set, a checkpoint is taken after each SEND, SENDEDI, or PUTMEMBER command.
- While receiving data for a RECEIVE or RECEIVEEDI command, if Information Exchange requests a checkpoint
- At the end of each RECEIVE or RECEIVEEDI command

The COMMIT command is valid only for user-initiated recovery. The TRANSMIT profile command determines the type of recovery. To specify user-initiated recovery, ensure that the RECOVERY parameter of TRANSMIT is set to u.

On restart, data transmission resumes after the last successful checkpoint or COMMIT command. If the error occurred while processing the first COMMIT command in the message command file and a checkpoint has not occurred, Information Exchange does not deliver any data and does not delete received data from the mailbox. In this case, Expedite Base/MVS retransmits all data upon restart.

Syntax

The COMMIT command has no parameters. Type the COMMIT command as follows:

```
commit;
```

DEFINEALIAS command

Use the DEFINEALIAS command to:

- Define a new alias
- Redefine an existing alias
- Change or delete an existing alias table

NOTE: Your user ID must be authorized to update the alias table, or you will receive a system error message in your mailbox.

Although you can generally specify parameters in any order, the DEFINEALIAS command entries must include an ALIASTABLE parameter and one of the following groups of parameters:

- ACCOUNT and USERID
- SYSID, ACCOUNT, and USERID
- ALIAS and ALIASNAME

You must pair these entries correctly. You must specify a DEFINENAME parameter with the ACCOUNT and USERID parameters or the SYSID, ACCOUNT, and USERID parameters. Another option is to specify the DEFINENAME parameter with the ALIAS and ALIASNAME parameters.

You may not specify another DEFINENAME parameter unless the definition for the previous DEFINENAME has been specified.

If the value of the FUNCTION parameter is e, you do not need to specify a DEFINENAME parameter.

NOTE: Do not define an alias more than once in a session, or the first alias will be overwritten.

Syntax

The dots between lines indicate that you can list as many accounts and aliases as necessary:

```
definealias

aliastable(define alias table) function(a|n|c|d|e)
authority(p|a|g)
definename(define aliasname 1)

account(account 1) userid(userid 1)
   or
sysid(system ID 1) account(account 1) userid(user ID 1)
   or
alias(alias 1) aliasname(aliasname 1)
     :    :
definename(define aliasname n)

account(account n) userid(userid n)
   or
sysid(system ID n) account(account n) userid(user ID n)
   or
alias(alias n) aliasname(aliasname n);
```

Parameters

aliastable
Alias table type and table name.

G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

function
Type of operation you want Expedite Base/MVS to request for the alias table.

a	Add the following entries into the alias table. This is the default. Change the following entries in the alias table.
d	Delete the following entries from the alias table. If Information Exchange deletes all the names from the alias table, you get the same results as with option **e**.
e	Erase the entire alias table.
n	Create a new alias table using the name specified in the ALIASTABLE parameter.

authority
Authorization level of the alias table being referenced. Authority is valid only if FUNCTION is ‘n’. You cannot change the authority of an existing table.

p	The alias table can be updated only by the owner of the table. This is the default.
a	The alias table can be updated by any administrator in the account.
g	The alias table can be updated by any user.

definename
Alias name that you wish to define in the table specified by ALIASTABLE. Use 1 to 16 alphanumeric characters.

sysid
System ID of an Information Exchange user. You need the system ID only if you specify the ACCOUNT and USERID parameters for a user on another Information Exchange system. You do not need to specify the system ID if you and your trading partner are on the same Information Exchange system. Use 1 to 3 alphanumeric characters.

account
Information Exchange account name. Use 1 to 8 alphanumeric characters.

userid
Information Exchange user ID. Use 1 to 8 alphanumeric characters.

alias
You can use the DEFINEALIAS command to define an alias that points to an alias name in another table. Indicate the table type and table name of this alias table.

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

aliasname
If you have specified an alias name in the ALIAS parameter, you must indicate the alias name in the table specified by ALIAS. Use 1 to 16 alphanumeric characters.

DEFINEALIAS command examples

The following example contains valid DEFINEALIAS entries:

```
definealias
aliastable(GX01) function(A) authority(G)
definename(DUNS01) account(ACT1) userid(USER1)
definename(DUNS02) sysid(EUR) account(ACT2) userid(USER2)
definename(DUNS03) alias(GX02) aliasname(DUNS51);
```

The following example shows invalid DEFINEALIAS entries:

```
definealias
aliastable(GX01) function(A) authority(G)
definename(DUNS01) account(ACT1) (INVALID entry: no userid)
definename(DUNS02) account(ACT2) userid(USER2)
userid(USER1) (INVALID entry: 2 userids)
definename(DUNS03) alias(GX02) aliasname(DUN51);
```

END command

Use the END command to end the Information Exchange session.

Syntax

The END command has no parameters. Type the END command as follows:

```
end;
```

NOTE: Specifying an END command in the command file results in an error if you use y as the AUTOEND value in the profile.

GETMEMBER command

Use the GETMEMBER command to copy a member from an existing Information Exchange library to an Information Exchange mailbox. When the member is available in the mailbox, specify a RECEIVE command as the next command in the input file to receive this member. You should use the RECEIVE or RECEIVESTREAM command as usual to receive the data from the mailbox to your system. If the destination is left blank on the GETMEMBER command, the default will be your own mailbox. The MSGNAME, MSGSEQNO, and CLASS parameters, if not specified, will default to values indicated when the member was stored in the library.

The library member requested may not be immediately available in your Information Exchange mailbox. If you do not receive the member and the GETMEMBER command completed successfully, try to receive the member in a subsequent session.

NOTE: You know the command completed successfully if there was a RETURN(00000) in the response file, OUTMSG.

Syntax

```
getmember

owner(library account)
library(library name)
member(member name)

account(account) userid (userID)
   or
sysid(system ID) account(account) userid(user ID)
   or
alias(alias) aliasname(alias name)
   or
listname(list name)

msgname(message name) msgseqno(message sequence number)
class(class) charge(1|3|5|6)
ack(blank|a|b|c|d|e|f|r) retain(retain);
```

Parameters

owner
Owner of the account of the requested library. The default is your own account. Use 1 to 8 alphanumeric characters.

library
Library from which the member is to be copied. Use 1 to 8 alphanumeric characters.

member
Library member to be copied. Use 1 to 8 alphanumeric characters.

sysid
System ID of an Information Exchange system user receiving the member. You need the system ID only if you specify the ACCOUNT and USERID parameters for a user on another Information Exchange system. If you specify a SYSID parameter, you must specify the ACCOUNT and USERID parameters. Use 1 to 3 alphanumeric characters.

account
Account name of the user receiving the member. Expedite Base/MVS uses this field together with the USERID parameter to identify the user. If you specify an ACCOUNT parameter, you must specify a USERID parameter. Use 1 to 8 alphanumeric characters. The default is your own account.

userid
User ID of an Information Exchange user receiving the member. Expedite Base/MVS uses this field together with the ACCOUNT parameter to identify the user. If you specify a USERID parameter, you must specify an ACCOUNT parameter. Use 1 to 8 alphanumeric characters. The default is your own user ID.

alias
Table type and table name of an alias table. This field is used in conjunction with the ALIASNAME parameter to identify the user receiving the file.

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

aliasname
Alias name that defined in the alias table. This field is used in conjunction with the ALIAS parameter to identify the user receiving the file. Use 1 to 16 alphanumeric characters.

listname
Name of a previously defined list of accounts and user IDs. This field is used to identify a list of users receiving the file. Use 1 to 8 alphanumeric characters.

msgname
Name you specify for the file. Use 1 to 8 alphanumeric characters.

msgseqno
Number assigned by you as a message control number for the data. Use 1 to 5 alphanumeric characters.

class
Specific name for the data. A receiver can use this name only to receive files of this class. The default is **blank**.

charge
Indicates how the sender wants to pay the file charges.

1	The receiver pays all charges.
3	The receiver pays all charges, if agreed to by the receiver. If not, the library owner and the receiver split the charges, if agreed to by the receiver. Otherwise, the library owner pays all charges.
5	The library owner pays the receive charge, if agreed to by the library owner. Otherwise, the receiver pays. This is the default.
6	The library owner pays the receive charge.

ack
Type of acknowledgment you want to receive.

blank	No acknowledgments created. This is the default.
a	Information Exchange creates only purge acknowledgments.
b	Information Exchange creates both receipt and delivery acknowledgments.
c	Information Exchange creates both receipt and purge acknowledgments.
d	Information Exchange creates only delivery acknowledgments. It then queues them to the user ID when Information Exchange delivers the data to a destination user ID and reaches a subsequent recovery point with the same destination user ID.
e	Information Exchange creates either purge or delivery acknowledgments.
f	Information Exchange creates receipt acknowledgments and either purge or delivery acknowledgments.
r	Information Exchange creates only receipt acknowledgments when Expedite Base/ MVS finishes sending a file.

NOTE: Expedite Base/MVS sends the receipt acknowledgment to whoever is paying for the GETMEMBER request. This is either the library owner or the individual who requested the member (issued the GETMEMBER command).

retain
Indicates the number of days Information Exchange keeps the file in the mailbox if no one receives it. Valid values are **blank** and **0** through **180**.

The maximum retention period and the default retention period can be different on different Information Exchange systems. These periods are system-dependent. The default retention period is determined when Information Exchange is installed. For installations in the U.S., the default retention period is 30 days. Contact your marketing representative for more information on these values.

If you specify **0** or **blank**, Information Exchange retains the file for the default retention period. If you specify a retention period that is longer than the maximum retention period for your system, Information Exchange retains the file for the default retention period.

GETMEMBER command example

```
GETMEMBER OWNER(IBM1) LIBRARY(MVSB45) MEMBER(JN00616)
CLASS(SMPPTFIN)CHARGE(6);
```

Results: This command puts member JN00616 from library MVSB45 into the user's mailbox with a class of SMPPTFIN.

LIST command

Use the LIST command to create a list of account and user IDs to use when sending and receiving files.

Syntax

The dots between lines indicate that you can list as many accounts and aliases as necessary:

```
list
listname(list name) function(a|d|e|n) listtype(p|a|g|t)
account(account 1) userid(userid 1)
     :    :
alias(alias 1) aliasname(aliasname 1)
     :    :
alias(alias n) aliasname(aliasname n)
     :    :
sysid(system ID n) account(account n) userid(user ID n);
```

A LIST command can include as many ALIAS and ALIASNAME entries or ACCOUNT and USERID entries as required to create the list. Although you can generally specify parameters in any order, the LIST command has the following limitations:

- LIST command entries must include either an ACCOUNT and USERID or an ALIAS and ALIASNAME. You must pair these entries correctly. For example, you must specify an ACCOUNT parameter with a USERID parameter.
- If you specify the SYSID parameter, you must specify it either before or between the ACCOUNT and USERID parameters to which it belongs. It cannot follow the ACCOUNT and USERID parameters for that system ID.
- At least one list entry is required for functions a (add entries), d (delete entries), or n (new list). No entries are permitted for function e (erase entire list).

ATTENTION: If you use the LIST command with session-level recovery, do not use the same distribution list name more than once in the same session. If you do, Information Exchange replaces the existing list definitions with the new list definitions. Therefore, any file you send uses the most recent list that you define.

Parameters

listname
Name of the list to process. This parameter is required when you use the LIST command. Use 1 to 8 alphanumeric characters.

function
Type of operation you want Expedite Base/MVS to perform on the list.

a	Add the following entries to the list.
d	Delete the following entries from the list. If Expedite Base/MVS deletes all the names from the list, you get the same result as with function e.
e	Erase the entire list.
n	Create a new list or replace a prior list with the same name specified in the LISTNAME parameter. This will replace an existing list with the same name. This is the default.

listtype
Type of list being referenced.

p	The list is a permanent, private list.
a	The list is a permanent list accessible to all members of your account.
g	The list is a permanent list with account level grouping. This type of list is used to limit communication with other users. Use it with care. For more information on this type of list, see Using Information Exchange Administration Services.
t	The list is temporary. It goes away when your session ends. This is the default.

account
Information Exchange account name. Use 1 to 8 alphanumeric characters.

userid
Information Exchange user ID. Use 1 to 8 alphanumeric characters.

alias
Table type and table name of an alias.

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

You create and maintain alias tables by using Information Exchange Administration Services (see Using Information Exchange Administration Services), or by using the DEFINEALIAS command (see “DEFINEALIAS command” on page 86).

aliasname
Alias name defined in the alias table. Use 1 to 16 alphanumeric characters.

sysid
System ID of an Information Exchange user. You need the system ID only if you specify the ACCOUNT and USERID parameters for a user on another system. You do not need to specify the system ID if you and your trading partner are on the same system. Use 1 to 3 alphanumeric characters.

LIST command example

```
list listname(mylist) function(n)
account(xxx1) userid(user01)
sysid(xxx) account(xxx1) userid(user02)
alias(gtbl) aliasname(alias1)
account(xxx2) userid(user02);
```

LISTLIBRARIES command

Use the LISTLIBRARIES command to request a list of Information Exchange libraries. Expedite Base/MVS returns a list of libraries to which you have access and which meet the criteria you specified on the command.

The message response records associated with the LISTLIBRARIES command are the LIBRARYLIST and RETURN records.

Syntax

```
listlibraries

authority(w|r)
selection(a|c)
owner(library owning account);
```

Parameters

authority
Indicates the user's access authority for the list of libraries requested.

w	Write or update access authority. This is the default.
r	Read access authority.

Expedite Base/MVS ignores this parameter if you use SELECTION(C) on this command.

selection
Indicates the level of library search.

a	A list of libraries with a specific owning account. This is the default.
c	A complete list of libraries.

owner
Indicates the owning account to use when SELECTION(A) is specified. The default is the user's account. Expedite Base/MVS ignores this parameter if you use SELECTION(C) on this command.

LISTLIBRARIES command example

```
listlibraries;
```

Results: Expedite Base/MVS returns a list of all libraries to which you have access.

LISTMEMBERS command

Use the LISTMEMBERS command to receive a list of members within an Information Exchange library.

The message response records associated with the LISTMEMBERS command are the MEMBERLIST and RETURN records.

Syntax

```
listmembers

owner(library owning account)
library(library name);
```

Parameters

owner
Indicates the library owning account. The default is the user's account.

library
Indicates the library containing the members to be listed. You must have read access to the library.

LISTMEMBERS command example

```
listmembers library(mylib);
```

Results: Expedite Base/MVS returns a list of members within the library *mylib* to which you have access.

LISTVERIFY command

Use the LISTVERIFY command to obtain information about lists. Information Exchange places the responses to the LISTVERIFY command in your Information Exchange mailbox as messages from the *SYSTEM* account and *LSTRSP* user ID, with a user class of LISTS.

Syntax

```
listverify
listname(list name) function(l|d|c|a|r|s|b) charge(1|2|3|4|5|6);
```

Parameters

listname
Name of a previously defined list that you want verified. Use 1 to 8 alphanumeric characters.

function
Type of operation you want the Information Exchange to perform. This parameter is required.

NOTE: The LISTNAME parameter is required if you select any function other than **a** and **d**.

l — List the users in a private, permanent list. Provide the name of the list in the LISTNAME parameter. The message is built with 20-character destination IDs. The format is alphanumeric, left-justified, and padded on the right with blanks. Each destination ID must occupy 20 positions, as follows:

- **Character 1 (coded value).** This indicates whether the address is a true Information Exchange address or an alias ID entry, as follows:

g	An alias table reference to a global table
o	An organization (account) alias table
p	A private table
i	An intersystem address
blank	A reference to a true Information Exchange address

- **Characters 2 through 4 (alphanumeric).** If character 1 is g, o, or p, this field names the alias table to be used to search for the name given in characters 5 through 20. If character 1 is i, this is the system identifier. If character 1 is blank, these characters are not used.
- **Characters 5 through 20 (alphanumeric)**. If character 1 is g, o, or p, this field specifies the 16-character alias name. If character 1 is blank or i, this field specifies the Information Exchange account (characters 5 through 12) and user ID (characters 13 through 20) of the desired destination.

NOTE: The LISTNAME parameter is required if you select this option.

d — List all private, permanent lists belonging to you. Information Exchange builds a message that consists of 8-character list names.

NOTE: The LISTNAME parameter is not valid with FUNCTION(D) specified.

c List the users in an account or group level list. Provide the name of the list in the LISTNAME parameter.

Information Exchange builds a message that consists of 20-character user IDs in the same syntax as that used in the FUNCTION(L) parameter (alias type, table name, or Information Exchange address).

NOTE: The LISTNAME parameter is required if you select this option.

a List all of the permanent lists belonging to your account. Information Exchange builds a message that consists of 8-character list names.

NOTE: The LISTNAME parameter is not valid with FUNCTION(A) specified.

r Verify that you can receive messages from all users in the list provided by the LISTNAME parameter. The user IDs in the list are checked to find out whether they:

- Are shown in the Information Exchange directory
- Have been authorized to send messages to your user ID

NOTE: The LISTNAME parameter is required if you select this option. Information Exchange creates system error messages for any user IDs in the list from which you cannot receive. If you can receive messages, you do not receive a response in your mailbox.

s Verify that you can send messages to all users in the list provided by the LISTNAME parameter. The user IDs in the list are checked to see whether or not they:

- Are valid user IDs of registered Information Exchange users
- Have been authorized to receive messages from your user ID

NOTE: The LISTNAME parameter is required if you select this option. Information Exchange creates system error messages for any user IDs in the list to which you cannot send. If you can send messages, you do not receive a response in your mailbox.

b Verify that you can both send messages to and receive messages from all users in the list provided by the LISTNAME parameter.

NOTE: The LISTNAME parameter is required if you select this option. Information Exchange creates system error messages for any user IDs in the list from which you cannot receive and to which you cannot send. If you can send and receive messages, you do not receive a response in your mailbox.

charge
If you want to have charge authority verified, specify the charge authority here. When sending messages, specify the same CHARGE parameter that you plan to use in the SEND, SENDEDI, or SENDSTREAM command. When receiving messages, specify the same CHARGE parameter that your trading partner plans to use in the SEND, SENDEDI, or SENDSTREAM command. If you do not specify this parameter, Expedite Base/MVS does not verify charge authority. Use 1 numeric character.

1	The receiver pays all charges.
2	The receiver pays all charges, if agreed to by the receiver. Otherwise, the sender and receiver split the charges.

3	The receiver pays all charges, if agreed to by the receiver. If not, the sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges.
4	The sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges.
5	The sender and receiver split the charges.
6	The sender pays all charges.

LISTVERIFY command example

```
listverify listname(mylist) function(b);
```

Results: Expedite Base/MVS verifies that you can both send and receive messages from all users in *mylist*.

MSGINFO command

Use the MSGINFO command to obtain information about messages that are currently available in your mailbox. Expedite Base/MVS places a MSGRESP record in the response file as a result of the MSGINFO command. The MSGRESP record contains summary information about available messages. You can obtain detailed information by using the QUERY command.

Syntax

The MSGINFO command has no parameters. Enter it as follows:

```
msginfo;
```

PURGE command

Use the PURGE command to delete a specific file from your Information Exchange mailbox. To authorize use of this command, the Service Administrator must use Information Exchange Administration Services to set the “Use message purge command” field to **y** in your Information Exchange profile.

The message response record associated with the PURGE command is the RETURN record.

Syntax

```
purge
msgkey(message key);
```

Parameters

msgkey
Indicates the 20-character message identifier of the file to be deleted. This message key can be obtained from the AVAILABLE record for this file in response to a QUERY command.

PURGE command example

```
purge msgkey(abc1de2fa34bcdef5a6b);
```

Results: Expedite Base/MVS deletes the file having a message key of *abc1de2fa34bcdef5a6b* from your mailbox.

PUTMEMBER command

Use the PUTMEMBER command to add a member to an existing Information Exchange library.

NOTE: You must have authority to update this library or you will receive a system error message in your mailbox.

Syntax

```
putmember

owner(library owner account)
library(library name)
member(member name)
fileid(file ID)
replace(n|y)
format(n|y) class(class) ack(blank|d)
msgname(message name) msgseqno(message sequence number)
datatype(e|b) truncate(n|y)
delimit(n|c|l) crlfeof(crlfeof)
verify(n|y) description(description)
destfile(destination file name)
destloc(destination file location);
```

Parameters

library
Name of the library to update. Use 1 to 8 alphanumeric characters.

member
Name of the member as it should appear in the library. Use 1 to 8 alphanumeric characters.

owner
Library owner's account. The default is your account. Use 1 to 8 alphanumeric characters.

fileid
File to be sent. Begin the file ID with dd: to specify a ddname; otherwise, Expedite Base/MVS assumes that the file ID is a data set name. Use 1 to 54 alphanumeric characters.

replace
Option you can use to replace a member with the same name as an existing member in the library.

y	Replace the member with the same name as an existing member in the library.
n	Do not replace the member with the same name as an existing member in the library. This is the default.

format
Option you can use to specify whether you want the data sent as a file or formatted as a free-format message.

y	Format the data as a free-format message. This implies fixed 79-byte records. Expedite Base/MVS pads records with blanks. You cannot specify DATATYPE(b), DELIMIT(c), or DELIMIT(l) with this option.
n	Send the data without free-format message formatting. This is the default.

class
Specific name for the data. A receiver can use this name to receive only files of this class. If you specify FORMAT(y), Expedite Base/MVS defaults to **ffmsg001**; otherwise, the Expedite Base/ MVS defaults to **blank**. Use 1 to 8 alphanumeric characters.

ack
Type of acknowledgment you want to receive.

blank	No acknowledgments created. This is the default.
d	Information Exchange creates only delivery acknowledgments. It then queues them to the user ID when Information Exchange delivers the data to the library and reaches a subsequent recovery point with the same destination user ID.

msgname
Name you specify for the file. Use 1 to 8 alphanumeric characters.

msgseqno
Number you assign to the data as a message control number. Use 1 to 5 alphanumeric characters.

datatype
Option you can use to indicate whether the data is EBCDIC or binary. ASCII systems translate EBCDIC data to ASCII when they receive it from Information Exchange.

e	EBCDIC data. This is the default.
b	Binary data.

truncate
Option you can use to indicate whether you want the Expedite Base/MVS to remove trailing blanks from each record before sending them to Information Exchange.

y	Truncate trailing blanks.
n	Do not truncate trailing blanks. This is the default.

delimit
Option you can use to specify how to preserve the record structure of the file. Expedite Base/ MVS inserts the appropriate delimiter type into the data and puts the appropriate information into the CDH for the receiver's use.

c	You want the records delimited by CRLF characters. Expedite Base/MVS inserts X'0D0A' at the end of each record to indicate the record structure, and ends the file with X'1A.' You can change the CRLFEOF characters with the CRLFEOF parameter.
l	You want the records delimited with delimiters that are 2 bytes long. Expedite Base/ MVS inserts a 2-byte length at the beginning of each record to indicate the record structure.
n	You want to send the records with no record delimiters. This is the default.

crlfeof
Option you can use to redefine the carriage-return, line-feed, and end-of-file characters used with the DELIMIT(c) option. You need this parameter when sending to a PC that receives with a translate table other than the standard Information Exchange translate table. Use 6 hexadecimal characters. The default value is **0D0A1A**.

verify
Option you can use to verify that the library is defined and that you have update access before sending the file. You cannot verify a library if it resides on another system. When you use the VERIFY parameter on the PUTMEMBER command, you are not charged for the verification request.

y	Verify that the library is defined and that you have update access before sending the file. If you cannot verify the library for either reason, the file is not sent and there is a WARNING record in the output file for this command.
n	Do not verify that the library is defined or that you have access. This is the default.

description
Provides a free-format description of the file. Use 1 to 79 alphanumeric characters. The description is only available to receiving interfaces that support the CDH. For more information on the CDH, see Appendix B, “Common data header.”

destfile
Indicates the file name to use in the Common Data Header (CDH) as the original file name. If the receiver is using a workstation-based Expedite and specifies ORIGFILE(Y) on the RECEIVE or RECEIVEEDI command, then Expedite uses this file name to store the data when it is received. If you specify a file name that is not valid on the receiver's system, then Expedite uses the file name specified by the receiver in the FILEID parameter on the RECEIVE or RECEIVEEDI command. By default, Expedite determines the original file name from the FILEID parameter on the PUTMEMBER command. Use 1 to 54 characters.

destloc
Indicates the file location to use in the Common Data Header (CDH). When the file is received or the receiver's mailbox is queried, Expedite shows this value in the RECEIVED or AVAILABLE record in the SENDERLOC parameter. By default, Expedite uses the file location on the sender's system. Use 1 to 65 characters.

NOTES:

1. Expedite Base/MVS users who want to implement the DESTFILE and DESTLOC feature must be at an Expedite Base Version 4, Release 3 or higher product on both the sending and receiving systems.

2. Expedite Base/MVS does not support the ORIGFILE parameter on the RECEIVE or RECEIVEEDI commands. Users on this system must allocate the receive file before running Expedite, and must specify the file name on the RECEIVE or RECEIVEEDI commands.

PUTMEMBER command example

```
putmember library(user1lib) member(mem1) fileid(user01.price.file);
```

QUERY command

The QUERY command returns a list of all files in your Information Exchange mailbox. Expedite Base/MVS places AVAILABLE records in the response file in response to the QUERY command. It writes an AVAILABLE record for each file in your mailbox.

Syntax

```
query
cdh(y|n);
```

Parameters

cdh
Indicates whether or not you want the CDH information included in the response.

y	Include the CDH information in the response. This is the default.
n	Do not include the CDH information in the response.

QUERY command example

```
query;
```

RECEIVE command

The RECEIVE command retrieves files from the Information Exchange mailbox.

Syntax

```
receive

alias(alias) aliasname(alias name)
   or
sysid(system ID) account(account) userid(user ID)
   or
account(account) userid(user ID)
   or
listname(list name)
   or
requeued(n|y)

fileid(file ID) format(n|y) class(class)
archiveid(archive ID) autoedi(n|y) ediopt(n|y)
delimited(c|l|n) dlmoverride(y|n) endstr(endstr)
resrecl(e|s) allfiles(y|n) nonedionly(n|y) msgkey(message key)
startdate(starting date) starttime(starting time)
enddate(ending date) endtime(ending time) timezone(l|g);
```

NOTE: If you are using supported data compression software and receive compressed data, not all parameters are supported. For more information, see Appendix E, "Using data compression."

Parameters

alias
Table type and table name of an alias table.

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where *xxx* identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where *xxx* identifies a 1 to 3-character name.
P*xxx*	Private alias table, where *xxx* identifies a 1 to 3-character table name.

NOTE: You create and maintain alias tables by using Information Exchange Administration Services (see Using Information Exchange Administration Services), or by using the DEFINEALIAS command (see "DEFINEALIAS command" on page 86).

aliasname
Alias name defined in the alias table. Use 1 to 16 alphanumeric characters.

sysid
System ID of a single-source user ID. Use 1 to 3 alphanumeric characters.

account
Account name of a single-source user ID. Use 1 to 8 alphanumeric characters.

userid
Source user ID. Use 1 to 8 alphanumeric characters.

listname
Name of a previously defined list of account and user IDs. Use 1 to 8 alphanumeric characters.

requeued
Option that specifies whether or not Expedite Base/MVS receives only files that have been retrieved from archive.

n	Receive all files in your mailbox. This is the default.
y	Receive files retrieved from archive only. If you specify y, you cannot specify an Information Exchange source.

fileid
Name of the data set or the ddname where the received data should go. Begin the file ID with dd: to specify a ddname; otherwise, Expedite Base/MVS assumes that the file ID is a data set name. This parameter is required. Use 1 to 54 alphanumeric characters.

format
Option that specifies whether you want to receive the data as a file or formatted as a free-format message. See "Free-format messages" on page 139 for more information.

NOTE: Expedite Base/MVS ignores this option when processing EDI or record delimiters.

y	Expedite Base/MVS formats the data as a free-format message. This implies fixed 79-byte records.
n	Expedite Base/MVS does not format the data as a free-format message. This is the default.

class
Message classification of the files to receive. You can limit the files received to those with the value specified in this field. Use 1 to 8 alphanumeric characters.

You can use a question mark as a wildcard character to substitute for any character or characters. For example, to select all messages whose user classification begins with AB1, type: **AB1?**

To select all those whose user classification ends with 999, type: **?999.**

If you specify FORMAT(y), the default value is **ffmsg001**, which is reserved for free-format messages. If you specify FORMAT(n), the default is **blank**, which indicates all classes.

archiveid
Archive reference identifier you want assigned to the files delivered in response to RECEIVE command requests. Use 1 to 8 alphanumeric characters.

autoedi
Option you can use to specify whether or not EDI processing as defined in Chapter 4, "Sending and receiving EDI data," should be done automatically for files if the CDH indicates that the files are EDI formatted.

y	Automatically perform EDI processing if the CDH indicates that the file is EDI formatted. This is the default.
n	Do not perform EDI processing for any of the files.

NOTE: This parameter does not do anything for files sent without a CDH. For more information on the CDH, see Appendix B, "Common data header."

ediopt
Option you can use to specify whether or not EDI data should be split at the end of EDI segments. This option is ignored unless the CDH indicates that the received message is EDI data and AUTOEDI is set to y.

y	Split records at the ends of EDI segments if the CDH indicates that the message is EDI formatted. This is the default.
n	Do not split records at the end of EDI segments.

delimited
Option you can use to determine the record structure of the received data. If you specify y for DLMOVERRIDE, Expedite Base/MVS uses the information in the DELIMITED parameter even if the CDH indicates the record is structured differently. If you specify n for DLMOVERRIDE, Expedite Base/MVS ignores the DELIMITED parameter if the CDH indicates the record structure of the data.

c	Split records at CRLF characters. If the CDH includes the CRLFEOF parameter, Expedite Base/MVS uses the CRLF characters from that parameter. Otherwise, Expedite Base/MVS uses X'0D0A' for the CRLF.
l	Split records according to the 2-byte length delimiters at the beginning of each record.
n	Store the data as it is received. The record length depends on the record length of the data set allocated to receive the data. This is the default.

dlmoverride
Option you can use to specify whether or not the DELIMITED parameter should override the structure specified by the CDH.

y	Format the data according to the DELIMITED parameter, even if the CDH indicates a record delimiter type.
n	If the CDH indicates a record delimiter type, format the data according to the CDH. This is the default.

endstr
Option you can use to specify the string used at the end of each message received. This permits you to separate messages within a file. Begin the delimiter string with s: to place the delimiter immediately after the received data, regardless of record boundaries. Begin the delimiter string with r: to place the delimiter at the beginning of the record, following the received data. The default is **r:**.

The s: and the r: are not considered part of the delimiter and are not placed in the received file. If you do not specify this parameter, or if you leave it blank, Expedite Base/MVS does not write a delimiter string at the end of each message. Use 1 to 79 alphanumeric characters.

resrecl
Indicates how Expedite Base/MVS should resolve a disparity between the length of the record built while receiving data and the logical record length of the data set allocated to receive data.

NOTE: If the data is processed as EDI data (using the AUTOEDI option), records are always split if they are too long for the data set. If the data does not contain delimiters, record length is determined by the logical record length of the response data set, and this does not apply.

e	The record received is too long for the data set, terminate the session with an error. This option is valid only for session-level recovery, because you might find it difficult to correct the error and still restart correctly using checkpoint-level recovery.
s	If the record received is too long for the data set, split the record and write a warning message to the response file. This is the default.

allfiles
Option you can use to specify whether or not Expedite Base/MVS should receive all files that match the RECEIVE specifications or just the first file in the Information Exchange mailbox that matches the RECEIVE specifications.

y	Receive all files that match the RECEIVE specifications. This is the default.
n	Receive only the first file that matches the RECEIVE specifications.

nonedionly
Option you can use to specify that only non-EDI data should be received.

y	Receive only those messages from your mailbox that are identified in the CDH as not having one of the EDI formats.
n	Receive all messages from your mailbox that satisfy your RECEIVE request. This is the default.

msgkey
Unique message key you can use to receive a specific file from the mailbox. Its value is taken from the AVAILABLE record in response to a QUERY command. Use 20 hexadecimal characters.

startdate
Starting date of a time range for the messages and files you want to receive from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 000102 (January 2, 1900).

If you limit the range to one day and an error occurs while sending the file, files could be left undetected in the mailbox. For example, assume that an error occurs while sending a file on June 11, 2004, and the file is not actually placed in the mailbox until 11:00 a.m. on June 12.

If you issue a RECEIVE command at 8:00 a.m. on June 12, 2004 with a range of 00:00:00 to 23:59:59 on June 11 specified, the file will not be received because it has not yet arrived in the mailbox. If you then issue another RECEIVE command at 8:00 a.m. on June 13, 2004, with a range of 00:00:00 to 23:59:59 on June 12 specified, the file will still not be received because its starting date of June 11 is outside of the specified range.

To avoid missing any files, you should either move the starting date earlier, or periodically check your mailbox.

starttime
Starting time of a time range for the files you want to receive from Information Exchange. For a file to qualify to be received, the time the file was sent to Information Exchange must fall within this time range. The format is *HHMMSS*. The default value is determined by Information Exchange, and is currently 000000 (0 hours).

enddate
Ending date of a time range for the messages and files you want to receive from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 420916 (September 16, 2042).

endtime
Ending time of a time range for files you want to receive from Information Exchange. For a file to qualify to be received, the time the file was sent to Information Exchange must fall within this time range. Use the *HHMMSS* format for the ending time. The default value for ENDTIME is determined by Information Exchange, and is currently 235347 (23:53:47).

timezone
The time reference in the STARTTIME and ENDTIME parameters.

l	Your local time, as specified by the TIMEZONE parameter of the IDENTIFY command. This is the default.
g	Greenwich mean time (GMT).

RECEIVE command example

```
receive fileid(user01.price.file) class(prices)
startdate(030701) starttime(000000)
enddate(031231) endtime(180000);
```

Results: Files in your Information Exchange mailbox with a user class of prices are received in *user01.price.file.* If more than one such file exists in your mailbox, it is appended to *user01.price.file*. Only files sent between 00:00:00 hours (your local time) on July 1, 2003, and 18:00:00 hours on December 31, 2003, are received from your mailbox.

RECEIVEEDI command

Use the RECEIVEEDI command to retrieve EDI-formatted files from the Information Exchange mailbox.

You can use the CLASS parameter to make sure you are receiving EDI data. To do so, you and your trading partner must agree on a name for the EDI data. Your trading partner must specify that name in the user CLASS parameter when sending the data, and you must specify that name in the CLASS parameter on the RECEIVEEDI command. The following example shows a RECEIVEEDI command using the CLASS parameter, which can be used to receive all the files with the class “editest” from the mailbox:

```
receiveedi fileid(edidata.filea) class(editest)
```

You can also use the EDIONLY parameter to receive the data marked in the CDH as EDI data. If your trading partner sent the data with an interface that supports the DFORMAT field in the CDH, then the CDH will be marked properly. When you use a RECEIVEEDI command with EDIONLY(Y) specified, Expedite Base/MVS does not receive files that are lacking a CDH.

Information Exchange interfaces before Release 3.0 do not support the CDH.

The following example shows a RECEIVEEDI command using the EDIONLY parameter. You can use this parameter to receive all the mailbox files that are marked in the CDH as EDI data:

```
receiveedi fileid(edidata.filea) edionly(y)
```

If you receive a file whose CDH indicates that the file is not EDI data, Expedite Base/MVS receives the file into records in the format indicated by the CDH. If a file has no CDH and is not recognizable as EDI data, Expedite Base/MVS receives the file without reformatting records. For more information on the CDH, see Appendix B, “Common data header.”

For more information on receiving EDI data, see “Receiving EDI data” on page 40.

Syntax

```
receiveedi

alias(alias) aliasname(alias name)
   or
sysid(system ID) account(account) userid(user ID)
   or
account(account) userid(user ID)
   or
listname(list name)
   or
requeued(n|y)

fileid(file ID) class(class)  archiveid(archive ID)
ediopt(n|y) allfiles(y|n) edionly(n|y)
msgkey(message key)
startdate(starting date) starttime(starting time)
enddate(ending date) endtime(ending time) timezone(l|g);
```

Parameters

alias
Table type and table name of an alias table.

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

aliasname
Alias name defined in the alias table. Use 1 to 16 alphanumeric characters.

sysid
System ID of a single-source user ID. Use 1 to 3 alphanumeric characters.

account
Account name of a single-source user ID. Use 1 to 8 alphanumeric characters.

userid
Source user ID. Use 1 to 8 alphanumeric characters.

listname
Name of a previously defined list of accounts and user IDs. Use 1 to 8 alphanumeric characters.

requeued
Option you can use to specify whether or not Expedite Base/MVS only receives files that have been retrieved from archive.

n	Receive any file in your mailbox. This is the default.
y	Receive files retrieved from archive only. If you specify y, you cannot specify an Information Exchange destination.

fileid
Name of the data set or the ddname where the received data should go. Begin the fileID with dd: to specify a ddname; otherwise, Expedite Base/MVS assumes that the file ID is a data set name. This parameter is required. Use 1 to 54 alphanumeric characters.

class
Option you can use to specify the message classification of the file you are receiving. You can limit the files received to those with the value specified in this field. You should always specify a class when receiving EDI files. Use 1 to 8 alphanumeric characters. The default value is **blank**, which indicates all classes.

If the EDI data was sent with a SENDEDI command using the default class, it arrives in your Information Exchange mailbox classified as follows:

#EE	EDIFACT data
#EU	UN/TDI data
#E2	X12 data
#EC	UCS data

You can use a question mark as a wildcard to substitute for any character or characters. For example, to receive only the EDI files sent with the default class, type: **#E?**

To select all EDI files with a class ending in 9, type: **?9**

archiveid
Archive reference ID to be assigned to the files delivered in response to RECEIVEEDI requests. Use 1 to 8 alphanumeric characters.

ediopt
Option you can use to specify whether or not Expedite Base/MVS should split EDI data at the end of EDI segments.

y	Split records at the end of EDI segments. If the CDH indicates that the information received is not EDI data, this option is ignored. This is the default.
n	Do not split records at the end of EDI segments. If the CDH indicates that the information received is not EDI data, this option is ignored.
f	Attempt to split records at the end of EDI segments, regardless of what the common data header (CDH) indicates. If the data is not valid EDI data, Expedite Base/MVS issues a warning.

NOTE: Use this option only if you are migrating from expEDIte/MVS Host to Expedite Base/MVS.

allfiles
Option you can use to specify whether Expedite Base/MVS should receive all files that match the RECEIVEEDI specifications or just the first file in the Information Exchange mailbox that matches the RECEIVEEDI specifications.

y	Receive all files that match the RECEIVEEDI specifications. This is the default.
n	Receive only the first file that matches the RECEIVEEDI specifications.

edionly
Option you can use to specify that only EDI data should be received.

y	Receive only those messages from your mailbox that are identified in the CDH as having one of the EDI formats.
n	Receive all messages from your mailbox that satisfy your RECEIVEEDI request. If a received file's CDH indicates that the files do not contain EDI data, Expedite Base/MVS receives the file into records, using the format indicated in the CDH. If a file has no CDH and is not recognizable as EDI data, Expedite Base/MVS receives the file without reformatting the records.

msgkey
The unique message key you can use to receive a specific file from the mailbox. Its value is taken from the AVAILABLE record in response to a QUERY command. Use 20 hexadecimal characters.

startdate
Starting date of a time range for the messages and files you want to receive from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 000102 (January 2, 1900).

If you limit the range to one day and an error occurs while sending the file, files could be left undetected in the mailbox. To avoid missing any files, you should either move the starting date earlier, or periodically check your mailbox. See the STARTDATE parameter on the RECEIVE command (page 106) for an example.

starttime
Starting time of a time range for the files you want to receive from Information Exchange. For a file to qualify to be received, the time the file was sent to Information Exchange must fall within this time range. The format is *HHMMSS*. The default value is determined by Information Exchange, and is currently 000000 (0 hours).

enddate
Ending date of a time range for the messages and files you want to receive from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 420916 (September 16, 2042).

endtime
Ending time of a time range for files you want to receive from Information Exchange. For a file to qualify to be received, the time the file was sent to Information Exchange must fall within this time range. Use the *HHMMSS* format for the ending time. The default value for ENDTIME is determined by Information Exchange, and is currently 235347 (23:53:47).

timezone
The time reference in the STARTTIME and ENDTIME parameters.

Your local time, as specified by the TIMEZONE parameter of the IDENTIFY command. This is the default. Enter g for Greenwich mean time (GMT).

RECEIVEEDI command example

```
receiveedi fileid(user01.orders) class(prices)
startdate(030701) starttime(000000)
enddate(031231) endtime(180000);
```

Results: Files in your Information Exchange mailbox with a user class of *prices* are received in *user01.orders*. If more than one such file exists in your mailbox, it is appended to *user01.orders*. Only files sent between 00:00:00 hours (your local time) on July 1, 2003, and 18:00:00 hours on December 31, 2003, are received from your mailbox.

RECEIVESTREAM command

Use the RECEIVESTREAM command to receive data directly into the message response file. When you receive data using the RECEIVESTREAM command, the data immediately follows the RECEIVED record in the response file. This command is valid only with session-level recovery.

Syntax

```
receivestream

alias(alias) aliasname(alias name)
   or
sysid(system ID) account(account) userid(user ID)
   or
account(account) userid(user ID)
   or
listname(list name)
   or
requeued(n|y)

endstr(end string) class(class) archiveid(archive ID)
allfiles(y|n) msgkey(message key)
startdate(starting date) starttime(starting time)
enddate(ending date) endtime(ending time) timezone(l|g);
```

NOTE: If you are using supported data compression software and receive compressed data, not all parameters are supported. For more information, see Appendix E, “Using data compression.”

Parameters

alias
Table type and table name of an alias table.

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

aliasname
Alias name defined in the alias table. Use 1 to 16 alphanumeric characters.

sysid
System ID of a single-source user ID. Use 1 to 3 alphanumeric characters.

account
Account name of a single-source user ID. Use 1 to 8 alphanumeric characters.

userid
Source user ID. Use 1 to 8 alphanumeric characters.

listname
Name of a previously defined list of accounts and user IDs. Use 1 to 8 alphanumeric characters.

requeued
Option you can use to indicate whether or not Expedite Base/MVS should receive only files that have been retrieved from archive.

n	Receive any file in your mailbox. This is the default.
y	Receive files retrieved from archive only. If you specify y, you cannot specify an Information Exchange destination.

endstr
Delimiter string that Expedite Base/MVS places in the response file following the received data.

Begin the delimiter string with s: to place the delimiter after the received data, regardless of record boundaries. Begin the delimiter string with r: to place the delimiter at the beginning of the record following the received data.

The default is **r:**. Expedite Base/MVS does not consider the s: and the r: part of the delimiter and does not place them in the response file. This parameter is required. Use 1 to 79 alphanumeric characters.

class
Message classification of the files to receive. You can limit the files received to those with the value specified in this field.

You can use a question mark as a wildcard to substitute for any character or characters at the beginning or end of a classification that you do not know.

For example, to select all messages whose classifications begin with AB1, type: **AB1?** To select all those whose user classifications end with 999, type: **?999.**

Use 1 to 8 alphanumeric characters. The default is blank, which indicates all classes.

archiveid
Archive reference identifier Expedite Base/MVS must assign to the data delivered in response to RECEIVESTREAM requests. Use 1 to 8 alphanumeric characters.

allfiles
Option you can use to specify whether Expedite Base/MVS should receive all data that meet the RECEIVESTREAM specifications or only the first item that matches the RECEIVESTREAM specifications.

y	Receive all data in the mailbox that matches the RECEIVESTREAM specifications. This is the default.
n	Receive only the first item that matches the RECEIVESTREAM specifications.

msgkey
Unique message key to receive a specific file from the mailbox. Use the QUERY command to get this value. Use 20 hexadecimal characters.

startdate
Starting date of a time range for the messages and files you want to receive from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 000102 (January 2, 1900).

If you limit the range to one day and an error occurs while sending the file, files could be left undetected in the mailbox. To avoid missing any files, you should either move the starting date earlier, or periodically check your mailbox. See the STARTDATE parameter on the RECEIVE command (page 106) for an example.

starttime
Starting time of a time range for the files you want to receive from Information Exchange. For a file to qualify to be received, the time the file was sent to Information Exchange must fall within this time range. The format is *HHMMSS*. The default value is determined by Information Exchange, and is currently 000000 (0 hours).

enddate
Ending date of a time range for the messages and files you want to receive from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. The default value is determined by Information Exchange, and is currently 420916 (September 16, 2042).

endtime
Ending time of a time range for files you want to receive from Information Exchange. For a file to qualify to be received, the time the file was sent to Information Exchange must fall within this time range. Use the *HHMMSS* format for the ending time. The default value for ENDTIME is determined by Information Exchange, and is currently 235347 (23:53:47).

timezone
The time reference in the STARTTIME and ENDTIME parameters.

l	Your local time, as specified by the TIMEZONE parameter of the command. This is the default.
g	Greenwich mean time (GMT).

RECEIVESTREAM command example

```
receivestream account(act3) userid(user005) endstr(xyzz) class(orders);
```

Results: Expedite Base/MVS places all files from account *act3* and user ID *user005* with a class of *orders* in your mailbox. Expedite Base/MVS places a delimiter of *xyzz* at the end of the received data.

SEND command

Use the SEND command to send a file to Information Exchange.

Syntax

```
send

alias(alias) aliasname(alias name)
   or
sysid(system ID) account(account) userid(user ID)
   or
account(account) userid(user ID)
   or
listname(list name)

fileid(file ID) format(n|y)
class(class) mode(blank|t) priority(blank|i|p)
charge(1|2|3|4|5|6)  ack(blank|a|b|c|d|e|f|r) msgname(message name)
msgseqno(message sequence number) datatype(e|b) truncate(n|y)
delimit(c|l|u|n) crlfeof(crlfeof string) verify(n|y|f)
description(description) retain(retain time) compress(n|y|t|v)
destfile(destination file name)
destloc(destination file location)
selectrcv(f|n|blank);
```

Parameters

alias
Table type and table name of an alias table:

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

NOTE: You create and maintain alias tables by using Information Exchange Administration Services (see Using Information Exchange Administration Services), or by using the DEFINEALIAS command. For more information, see “DEFINEALIAS command” on page 86.

aliasname
Alias name defined in the alias table. Use 1 to 16 alphanumeric characters.

sysid
System ID of an Information Exchange user to whom you are sending data. You need the system ID only if you specify the ACCOUNT and USERID parameters for a user on another Information Exchange system. If you specify a SYSID parameter, you must specify the ACCOUNT and USERID parameters. Use 1 to 3 alphanumeric characters.

account
Account of an Information Exchange user to whom you are sending data. Expedite Base/MVS uses this field together with the USERID parameter to identify the user. If you specify an ACCOUNT parameter, you must specify a USERID parameter. Use 1 to 8 alphanumeric characters.

userid
User ID of an Information Exchange user to whom you are sending data. Expedite Base/MVS uses this field together with the ACCOUNT parameter to identify the user. If you specify a USERID parameter, you must also specify an ACCOUNT parameter. Use 1 to 8 alphanumeric characters.

listname
Name of a previously defined list of account and user IDs. Use 1 to 8 alphanumeric characters.

fileid
File to be sent. Begin the file ID with dd:; otherwise, Expedite Base/MVS assumes that the file ID is a data set name. Use 1 to 54 alphanumeric characters.

format
Option you can use to specify whether or not you want the data sent as a file or formatted as a free-format message. For more information, see “Free-format messages” on page 139.

y	Format the data as a free-format message. This implies fixed 79-byte records. Expedite Base/MVS pads records with blanks. You cannot specify DATATYPE(b), DELIMIT(c), or DELIMIT(l) with this option.
n	Send the data without free-format message formatting. This is the default.

class
Specific name for the data. A receiver can use this name to receive only files of this class.

If you specify FORMAT(y), this defaults to **ffmsg001.** Otherwise, it defaults to **blank.** Use 1 to 8 alphanumeric characters.

mode
Option that indicates whether the file is a test file or a normal file.

blank	Normal file
t	Test-mode file

The default is blank.

priority
Class of delivery service for this file.

blank	Normal priority. This is the default.
i	Express delivery to those users who have continuous receive capability and are currently in session with Information Exchange. (Expedite Base/MVS does not support continuous receive capability.)
p	High priority.

charge
Option that indicates to Information Exchange how the sender wants to pay the file charges.

1	The receiver pays all charges.
2	The receiver pays all charges, if agreed to by the receiver. Otherwise, the sender and receiver split the charges.

3	The receiver pays all charges, if agreed to by the receiver. If not, the sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges. This is the default.
4	The sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges.
5	The sender and receiver split the charges.
6	The sender pays all charges.

ack
Type of acknowledgment you want to receive.

blank	No acknowledgments created. This is the default.
a	Information Exchange creates only purge acknowledgments.
b	Information Exchange creates both receipt and delivery acknowledgments.
c	Information Exchange creates both receipt and purge acknowledgments.
d	Information Exchange creates only delivery acknowledgments. It then queues them to the user ID when Information Exchange delivers the data to a destination user ID and reaches a subsequent recovery point with the same destination user ID.
e	Information Exchange creates either purge or delivery acknowledgments.
f	Information Exchange creates receipt acknowledgments and either purge or delivery acknowledgments.
r	Information Exchange creates only receipt acknowledgments when Expedite Base/ MVS finishes sending a file.

msgname
Name you specify for the file. Use 1 to 8 alphanumeric characters.

msgseqno
Number assigned by you as a message control number for the data. Use 1 to 5 alphanumeric characters.

datatype
Option you can use to indicate whether the data is EBCDIC or binary. ASCII systems translate EBCDIC data to ASCII when they receive it from Information Exchange. For more information on EBCDIC and binary data, see Chapter 3, “Communicating with other operating systems.”

e	EBCDIC data. This is the default.
b	Binary data.

truncate
Option you can use to indicate whether or not you want Expedite Base/MVS to remove trailing blanks from each record before sending them to Information Exchange.

y	Truncate trailing blanks.
n	Do not truncate trailing blanks. This is the default.

delimit
Option you can use to indicate how to preserve the record structure of the file. Expedite Base/MVS puts the appropriate delimiter type into the data and puts the appropriate information into the CDH for the receiver's use.

c	You want the records delimited by CRLF characters. Expedite Base/MVS inserts X'0D0A' at the end of each record to indicate the record structure. Expedite Base/MVS ends the file with X'1A'. You can change the CRLF characters with the CRLFEOF parameter.
l	You want the records delimited with delimiters 2 bytes long. Expedite Base/MVS inserts a 2-byte length at the beginning of each record to delineate the record structure.
n	You want to send the records with no record delimiters. This is the default.
u	You do not want delimiting characters added; the records you are sending are already delimited with CRLF characters, and you want the CDH to indicate this.

crlfeof
Option you can use to redefine the carriage-return, line-feed, and end-of-file characters used with the DELIMIT(c) option. You need this parameter when sending to a PC that receives with a translate table other than the standard Information Exchange translate table. Use 6 hexadecimal characters. The default value is **0D0A1A.**

verify
Option you can use to verify that the receiver exists before sending the file.

n	Do not verify that the receiver exists. This is the default.
y	Verify that the receiver exists before sending the data.
f	Verify that the receiver exists. If Information Exchange cannot tell whether the receiver exists (for example, if the receiver is on another system), send the data anyway.

If you request verification, you might incur an Information Exchange message charge. For more information on validations, see "Validating addresses, payment levels, and authorizations when sending data" on page 139.

description
Free-format description of the file sent. Use 1 to 79 alphanumeric characters. The description is available only to receiving interfaces that support the CDH. For more information on the CDH, see Appendix B, "Common data header."

retain
Indicates the number of days Information Exchange keeps the file in the mailbox if no one receives it. Valid values are **blank** and **0** through **180**.

The maximum retention period and the default retention period can be different on different Information Exchange systems. These periods are system-dependent. The default retention period is determined when Information Exchange is installed. For installations in the U.S., the default retention period is 30 days. Contact your marketing representative for more information about these values.

If you specify **0** or **blank**, Information Exchange retains the file for the default retention period. If you specify a retention period that is longer than the maximum retention period for your system, Information Exchange retains the file for the default retention period.

compress
Indicates whether the specified file should be compressed.

n	Do not compress the specified file. This is the default.
y	Compress the specified file.
t	For each sender/receiver pair, use the setting of the COMPRESS parameter (y or n) indicated in the CPLOOKUP table. Write a warning message to the response file, OUTMSG, if the receiver/sender pair is not in the CPLOOKUP table, and send the data as is.
v	For each sender/receiver pair, use the setting of the COMPRESS parameter (y or n) indicated in the CPLOOKUP table. Do not write a warning message to the response file, OUTMSG, if the receiver/sender pair is not in the CPLOOKUP table, and send the data as is.

You must have the supported compression software installed in order to use compression.

Some of the parameters of this command may not apply when sending compressed data. For more information, see Appendix E, "Using data compression."

destfile
Indicates the file name to use in the Common Data Header (CDH) as the original file name. If the receiver is using a workstation-based Expedite and specifies ORIGFILE(Y) on the RECEIVE or RECEIVEEDI command, then Expedite uses this file name to store the data when it is received. If you specify a file name that is not valid on the receiver's system, then Expedite uses the file name specified by the receiver in the FILEID parameter on the RECEIVE or RECEIVEEDI command. By default, Expedite determines the original file name from the FILEID parameter on the SEND, SENDEDI, or PUTMEMBER command. Use 1 to 54 characters.

destloc
Indicates the file location to use in the Common Data Header (CDH). When the file is received or the receiver's mailbox is queried, Expedite shows this value in the RECEIVED or AVAILABLE record in the SENDERLOC parameter. By default, Expedite uses the file location on the sender's system. Use 1 to 65 characters.

NOTES:

1. Expedite Base/MVS users who want to implement the DESTFILE and DESTLOC feature must be at an Expedite Base Version 4, Release 3 or higher product on both the sending and receiving systems.

2. Expedite Base/MVS does not support the ORIGFILE parameter on the RECEIVE or RECEIVEEDI commands. Users on this system must allocate the receive file before running Expedite, and must specify the file name on the RECEIVE or RECEIVEEDI commands.

selectrcv
Indicates whether or not the file should be marked in Information Exchange so that the file cannot be received on a blanket receive.

f	Marks the file in Information Exchange so that the file cannot be received on a blanket receive.
n	Allows the file to be received by a blanket receive. This is the default.
blank	Allows the file to be received by a blanket receive.

When **f** is selected in SELECTRCV, the file being sent is marked in Information Exchange and cannot be received by a blanket receive. The file can only be received if the receiver specifies one of the following:

- Sender ID (account ID and user ID)
- Message class
- Message key

SEND command example

```
send fileid(user04.test.file) alias(ptb3) aliasname(name)
   class(question);
```

Results: Expedite Base/MVS sends file *user04.test.file* to Information Exchange for alias *ptb3* and aliasname *name*. The file is assigned a class of *question*.

SENDEDI command

Use the SENDEDI command to send an EDI-formatted file to Information Exchange. The file can contain X12, UCS, EDIFACT, UN/TDI data, or any combination of these. For more information on sending EDI data, see Chapter 4, “Sending and receiving EDI data.”

Syntax

```
sendedi

fileid(file ID) mode(blank|t)
priority(blank|i|p) charge(1|2|3|4|5|6) ack(blank|a|b|c|d|e|f|r)
msgname(message name) msgseqno(message sequence number) class(class)
verify(n|y|f|c|g) description(description) retain(retain time)
compress(n|y|t|v)
destfile(destination file name)
destloc(destination file location)
selectrcv(f|n|blank);
```

Parameters

fileid
File that you want to send. Begin the file ID with dd:; otherwise, Expedite Base/MVS assumes the file ID is a data set name. Use 1 to 54 alphanumeric characters.

mode
Option you can use to indicate whether a file is a test or a normal file.

blank	Normal file. This is the default.
t	Test-mode file.

priority
Class of delivery service for this file.

blank	Normal priority. This is the default.
i	Express delivery to those users who have the continuous receive capability and are currently in session with Information Exchange. (Expedite Base/MVS does not support the continuous receive capability.)
p	High priority.

charge
Option you can use to indicate to Information Exchange how the sender wants to pay the file charges.

1	The receiver pays all charges.
2	The receiver pays all charges, if agreed to by the receiver. Otherwise, the sender and receiver split the charges.
3	The receiver pays all charges, if agreed to by the receiver. If not, the sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges. This is the default.
4	The sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges.

5	The sender and receiver split the charges.
6	Indicates the sender pays all charges.

ack
Type of acknowledgment files you want to receive.

blank	Information Exchange creates no acknowledgments. This is the default.
a	Information Exchange creates only purge acknowledgments.
b	Information Exchange creates both receipt and delivery acknowledgments.
c	Information Exchange creates both receipt and purge acknowledgments.
d	Information Exchange creates only delivery acknowledgments. It then queues them to the user ID when Information Exchange delivers the file to a destination user ID and reaches a subsequent recovery point with that destination user ID.
e	Information Exchange creates either purge or delivery acknowledgments.
f	Information Exchange creates receipt and either purge or delivery acknowledgments.
r	Information Exchange creates only receipt acknowledgments when Expedite Base/MVS finishes sending a file.

msgname
Name you specify for the file. For more information on the default values of MSGNAME, see "Message name (MSGNAME) assignment" on page 39. Use 1 to 8 alphanumeric characters.

msgseqno
Number assigned by you as a message control number for the data. For more information on the default values of MSGSEQNO, see "Message sequence number (MSGSEQNO) assignment" on page 39. Use 1 to 5 alphanumeric characters.

class
Unique name for this file. This name can be used by a receiver to receive only files of this class.

If you do not specify the class with either the CLASS parameter or the EDI envelope, the message class defaults as follows:

This EDI data type:	Uses this default message class:
X12	#E2
UCS	#EC
UN/TDI	#EU
EDIFACT	#EE

For more information on CLASS assignment, see "Message class (CLASS) assignment" on page 40. Use 1 to 8 alphanumeric characters.

verify
Option you can use to verify that the receiver exists before sending the file.

n	Do not verify that the receiver exists. This is the default.
y	Verify that the receiver exists before sending the data. If the verification fails or the destination cannot be verified, the envelope is not sent. Furthermore, if there are multiple envelopes in the file, envelopes following the one with the error are not sent.
f	Verify that the receiver exists before sending the data. If Information Exchange cannot tell whether the receiver exists (for example, if the receiver is on another system), send the data anyway. If the verification fails for an envelope destined for the same Information Exchange system as the sender, the envelope is not sent and if there are multiple envelopes in the file, envelopes following the one with the error are not sent.
c	Verify that the receiver exists before sending the data. If the verification fails or the destination cannot be verified, the envelope is not sent. If there are multiple envelopes in the file, continue processing those envelopes following the one in error.
g	Verify that the receiver exists before sending the data. If Information Exchange cannot tell whether the receiver exists (for example, if the receiver is on another system), send the data anyway. If verification fails for an envelope destined for the same Information Exchange system as the sender, the envelope is not sent. If there are multiple envelopes in the file, continue processing those envelopes following the one in error.

If you request verification, you might incur an Information Exchange message charge. For more information, see “Validating addresses, payment levels, and authorizations when sending data” on page 139.

description
Free-format description of the file sent. The description is only available to receiving interfaces that support the CDH. For more information on the CDH, see Appendix B, “Common data header.” Use 1 to 79 alphanumeric characters.

retain
Indicates the number of days Information Exchange keeps the file in the mailbox if no one receives it. Valid values are **blank** and **0** through **180**.

The maximum retention period and the default retention period can be different on different Information Exchange systems. These periods are system-dependent. The default retention period is determined when Information Exchange is installed. For installations in the U.S., the default retention period is 30 days. Contact your marketing representative for more information on these values.

If you specify **0** or **blank**, Information Exchange retains the file for the default retention period. If you specify a retention period that is longer than the maximum retention period for your system, Information Exchange retains the file for the default retention period.

compress
Indicates whether the specified file should be compressed.

n	Do not compress the specified file. This is the default.
y	Compress the specified file.

t	For each receiver/sender pair, use the setting of the COMPRESS parameter (y or n) indicated in the CPLOOKUP table. Write a warning message to the response file, OUTMSG, if the receiver/sender pair is not in the CPLOOKUP table and send the data as is.
v	For each receiver/sender pair, use the setting of the COMPRESS parameter (y or n) indicated in the CPLOOKUP table. Do not write a warning message to the response file, OUTMSG, if the receiver/sender pair is not in the CPLOOKUP table, and send the data as is.

You must have the supported compression software installed in order to use compression.

Some of the parameters of this command may not apply when you are sending compressed data. For more information, see Appendix E, "Using data compression."

destfile
Indicates the file name to use in the Common Data Header (CDH) as the original file name. If the receiver is using a workstation-based Expedite and specifies ORIGFILE(Y) on the RECEIVE or RECEIVEEDI command, then Expedite uses this file name to store the data when it is received. If you specify a file name that is not valid on the receiver's system, then Expedite uses the file name specified by the receiver in the FILEID parameter on the RECEIVE or RECEIVEEDI command. By default, Expedite determines the original file name from the FILEID parameter on the SEND, SENDEDI, or PUTMEMBER command. Use 1 to 54 characters.

destloc
Indicates the file location to use in the Common Data Header (CDH). When the file is received or the receiver's mailbox is queried, Expedite shows this value in the RECEIVED or AVAILABLE record in the SENDERLOC parameter. By default, Expedite uses the file location on the sender's system. Use 1 to 65 characters.

NOTES:

1. Expedite Base/MVS users who want to implement the DESTFILE and DESTLOC feature must be at an Expedite Base Version 4, Release 3 or higher product on both the sending and receiving systems.

2. Expedite Base/MVS does not support the ORIGFILE parameter on the RECEIVE or RECEIVEEDI commands. Users on this system must allocate the receive file before running Expedite, and must specify the file name on the RECEIVE or RECEIVEEDI commands.

selectrcv
Indicates whether or not the file should be marked in Information Exchange so that the file cannot be received on a blanket receive.

f	Marks the file in Information Exchange so that the file cannot be received on a blanket receive.
n	Allows the file to be received by a blanket receive. This is the default.
blank	Allows the file to be received by a blanket receive.

When **f** is selected in SELECTRCV, the file being sent is marked in Information Exchange and cannot be received by a blanket receive. The file can only be received if the receiver specifies one of the following:

- Sender ID (account ID and user ID)

- Message class
- Message key

SENDEDI command example

```
sendedi fileid(user-4.edi.data) class(question) retain(0);
```

Results: Expedite Base/MVS sends file *user04.edi.data* to Information Exchange with a class of *question*. The default retention period applies.

SENDSTREAM command

Use the SENDSTREAM command to send data from the message command file (INMSG) to Information Exchange. You place the data immediately after the SENDSTREAM command in the command file. This command is valid only with session-level recovery.

Syntax

```
sendstream

alias(alias) aliasname(alias name)
   or
sysid(system ID) account(account) userid(user ID)
   or
account(account) userid(user ID)
   or
listname(list name)

endstr(end string) datatype(e|b) mode(blank|t)
priority(blank|i|p) charge(1|2|3|4|5|6) ack(blank|a|b|c|d|e|f|r)
msgname(message name) msgseqno(message sequence number) class(class)
verify(n|y|f|c|g) description(description) retain(retain time)
compress(n|y|t|v)
selectrcv(f|n|blank);
```

Parameters

alias
Table type and table name of an alias table.

blank	An alias name was not used. This is the default.
G*xxx*	Global alias table, where xxx identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where xxx identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where xxx identifies a 1 to 3-character table name.

aliasname
Alias name that defined in the alias table. Use 1 to 16 alphanumeric characters.

sysid
System ID of a single destination user ID. You need SYSID only if you send the file to a user on another system. Use 1 to 3 alphanumeric characters.

account
Account name of a single destination user ID. Use 1 to 8 alphanumeric characters.

userid
Destination user ID. Use 1 to 8 alphanumeric characters.

listname
Name of a previously defined list of account and user IDs. Use 1 to 8 alphanumeric characters.

endstr
Character string used to mark the end of the data contained in the command stream. You can split the end string across record boundaries. Use 1 to 79 alphanumeric characters. This parameter is required.

datatype
Option you can use to indicate whether the data is EBCDIC or binary. ASCII systems translate EBCDIC data to ASCII when they receive it from Information Exchange. For more information on EBCDIC and binary data, see Chapter 3, “Communicating with other operating systems.”

e	EBCDIC data. This is the default.
b	Binary data.

mode
Option you can use to indicate whether the file is a test file or a normal file.

blank	Normal file. This is the default.
t	Test mode file.

priority
Class of delivery service for this data.

blank	Normal priority. This is the default.
i	Express delivery to those users who have the continuous receive capability and are currently in session with Information Exchange. (Expedite Base/MVS does not support the continuous receive capability.)
p	High priority.

charge
Option you can use to indicate how you would like the charges paid.

1	Receiver pays all charges.
2	Receiver pays all charges if agreed to. Otherwise, the sender and receiver split the charges.
3	Receiver pays all charges if agreed to; otherwise, the sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges. This is the default.
4	Sender and receiver split the charges, if agreed to by the receiver. Otherwise, the sender pays all charges.
5	Sender and receiver split the charges.
6	Sender pays all charges.

ack
Type of acknowledgment you want to receive.

blank	Information Exchange creates no acknowledgments. This is the default.
a	Information Exchange creates only purge acknowledgments.
b	Information Exchange creates both receipt and delivery acknowledgments.
c	Information Exchange creates both receipt and purge acknowledgments.

d	Information Exchange creates only delivery acknowledgments. It queues the delivery acknowledgments to the user ID when Information Exchange delivers the data to a destination user ID and reaches a subsequent recovery point with that destination user ID.
e	Information Exchange creates either a purge or a delivery acknowledgment.
f	Information Exchange creates receipt and either a purge or a delivery acknowledgment.
r	Information Exchange creates only receipt acknowledgments when Expedite Base/MVS finishes sending a file.

msgname
A user-specified name for the data. Use 1 to 8 alphanumeric characters.

msgseqno
You assign the MSGSEQNO to specify a message control number for the data. Use 1 to 5 alphanumeric characters.

class
Enables you to specify a unique name for this data. A receiver can then use this name to receive only files of this class. Use 1 to 8 alphanumeric characters.

verify
Option you can use to verify that the receiver exists before sending the data.

n	Do not verify that the receiver exists. This is the default.
y	Verify that the receiver exists before sending the data.
f	Verify that the receiver exists. If Information Exchange cannot tell whether the receiver exists (for example, if the receiver is on another system), send the data anyway.

If you request verification, you might incur an Information Exchange message charge. For more information, see “Validating addresses, payment levels, and authorizations when sending data” on page 139.

description
A free-format description of the data sent. Use 1 to 79 alphanumeric characters.

retain
Indicates the number of days Information Exchange keeps the file in the mailbox if no one receives it. Valid values are **blank** and **0** through **180**.

The maximum retention period and the default retention period can be different on different Information Exchange systems. These periods are system-dependent. The default retention period is determined when Information Exchange is installed. For installations in the U.S., the default retention period is 30 days. Contact your marketing representative for more information on these values.

If you specify **0** or **blank**, Information Exchange retains the file for the default retention period. If you specify a retention period that is longer than the maximum retention period for your system, Information Exchange retains the file for the default retention period.

compress
Indicates whether the specified file should be compressed.

n	Do not compress the specified file. This is the default.
y	Compress the specified file.
t	For each sender/receiver pair, use the setting of the COMPRESS parameter (y or n) indicated in the CPLOOKUP table. Write a warning message to the response file, OUTMSG, if the receiver/sender pair is not in the CPLOOKUP table, and send the data as is.
v	For each receiver/sender pair, use the setting of the COMPRESS parameter (y or n) indicated in the CPLOOKUP table. Do not write a warning message to the response file, OUTMSG, if the receiver/sender pair is not in the CPLOOKUP table, and send the data as is.

You must have the supported compression software installed in order to use compression.

Some of the parameters of this command may work differently when sending compressed data. For more information, see Appendix E, “Using data compression.”

selectrcv
Indicates whether or not the file should be marked in Information Exchange so that the file cannot be received on a blanket receive.

f	Marks the file in Information Exchange so that the file cannot be received on a blanket receive.
n	Allows the file to be received by a blanket receive. This is the default.
blank	Allows the file to be received by a blanket receive.

When **f** is selected in SELECTRCV, the file being sent is marked in Information Exchange and cannot be received by a blanket receive. The file can only be received if the receiver specifies one of the following:

- Sender ID (account ID and user ID)
- Message class
- Message key

SENDSTREAM command example

```
sendstream listname(list3) class(testclas) endstr(xxx);
```

Results: This data is sent to everyone who is defined in *list3 xxx*.

SESSIONINFO command

Use the SESSIONINFO command to obtain information about your Information Exchange session. Expedite Base/MVS places a SESSIONRESP record in the response file as a result of the SESSIONINFO command. It contains Information Exchange session information.

Syntax

The SESSIONINFO command has no parameters. Type the SESSIONINFO command as follows:

```
sessioninfo;
```

START command

Use the START command to begin an Information Exchange session. If Information Exchange successfully starts the session, Expedite Base/MVS returns a SESSIONKEY in the RETURN response record.

Syntax

```
start

account(account) userid(user ID)
iepassword(IE password) niepassword(new IE password) check(y|n);

keyringfile(fspec)
   OR
keyringfile(fspec) keyringpassword(password)
   OR
   keyringfile(fspec) keyringstashfile(fspec);
```

NOTE: If the AUTOSTART value in the profile is y, issuing a START command results in an error condition.

Parameters

account
Account name of the user. Use 1 to 8 alphanumeric characters. The default is the IEACCOUNT value used in the IDENTIFY command.

userid
User ID of the user. Use 1 to 8 alphanumeric characters. The default is the IEUSERID value used in the IDENTIFY command.

iepassword
Information Exchange password. Use 1 to 8 alphanumeric characters. The default is the IEPASSWORD value used in the IDENTIFY command.

niepassword
New Information Exchange password. The new password becomes effective at next successful Information Exchange session end. Use 1 to 8 alphanumeric characters. The default is the NIEPASSWORD value used in the IDENTIFY command.

NOTE: Remove this parameter and change the IEPASSWORD parameter after you change your password.

check
Determines whether or not the status of the previous session is checked.

y	Check the status of the previous session. Do not specify any other The status of the previous session will be indicated on the STARTED response record.
n	Do not check the status of the previous session; start a session as usual. This is the default.

keyringfile
The name of the key database HFS file or the RACF managed key ring. This parameter is required if ENABLESSL= y on the SSL command, and must be created prior to invoking Expedite Base/MVS. Use 2 to 251 characters.

If you specify a key database HFS file, you must provide either the password or the stash file that contains the password. For more information on creating a key database, see Chapter 11, “Communicating with Information Exchange using SSL.” You must fully qualify this name including the file extension. A fully qualified key database HFS file name must be between 2 and 251 characters, and should either have no extension or have an extension of **.kdb**. The maximum allowed length of a database name is 247characters excluding any extension characters. The key database name cannot end with **.rdb** or **.sth**.

If you specify a RACF managed key ring, no other parameters are necessary as access authority is controlled through RACF. The format for specifying a RACF managed key ring is userid/keyring.

The values in this parameter are case sensitive.

keyringpassword
The password for the key database. This value is established when the key database is created. Use 1 to 128 characters. If you use this parameter, do not use keyringstashfile.

keyringstashfile
The name of the key database password stash file. The name of this file must include a **.sth** extension. You must create this file prior to invoking Expedite Base/MVS.

START command example

```
start account(acct) userid(user01) iepassword(mypswd)
keyringfile(user01/ExpKeyRing);
```

TESTMSG command

The TESTMSG command tells Information Exchange to load one or more test messages into the mailbox. Test messages are from account *SYSTEM* and user ID *TSTMSG*. You must specify this account and user ID with the RECEIVE command to receive the test messages. If you do not specify the account and user ID, you cannot receive the test messages.

Syntax

```
testmsg

startmsg(0|1|2|3|4|5) endmsg(0|1|2|3|4|5);
```

Parameters

startmsg
The number of the first test message that Expedite Base/MVS must place in the mailbox.

0	Return the numeric characters, 0 through 9, in the test message.
1	Return the uppercase alphabetic characters in the test message.
2	Return the lowercase alphabetic characters in the test message.
3	Return the numeric characters, followed by uppercase alphabetic characters, followed by lowercase alphabetic characters in the test message.
4	Return the special characters in the test message.
5	Return all 256 characters in ascending sequence.

This parameter is required.

endmsg
The number of the last test message Expedite Base/MVS must place in the mailbox. Valid numbers are described under the STARTMSG parameter. The ENDMSG must be greater than or equal to the STARTMSG. If STARTMSG and ENDMSG are the same, Expedite Base/MVS returns a single message. This parameter is required.

TESTMSG command example

```
testmsg startmsg(0) endmsg(5);
```

Additional features of Expedite Base/MVS

You can use Expedite Base/MVS to retrieve audit data, create distribution lists, transfer free-format messages, validate addresses, payment levels, and authorizations, or work with libraries. This chapter describes the use of these features.

Using audit trails

Information Exchange provides an audit trail that you can retrieve using Expedite Base/MVS or view using Information Exchange Administration Services. For information on viewing audit data, see Using Information Exchange Administration Services. Audit trails are files that contain information about files you send and receive. Audit trails can include the following information:

- Account and user ID of the person who received the file
- Account and user ID of the person who sent the file
- Date and time the file was sent or received
- User class of the file
- Current status of the file (for example, delivery date and time)

You can use the information in audit trails to see the status or final disposition of a file. For example, if you send an order electronically to a manufacturer on July 1 and do not receive the items as expected, you can request an audit trail of the files you sent to the manufacturer on July 1. The audit trail may show that the manufacturer never received the order and it is still in their mailbox. If the audit trail shows that they received the order, you may want to check with the manufacturer to find out if the order was ever filled.

You can also use audit trails to see how many files you sent or received over a specific time period. For example, you can request an audit trail that shows all of the files you sent over the last three weeks.

Retrieving audit trails

Use the AUDIT command to retrieve an audit trail from Information Exchange and place the information in your mailbox. When audit data is available, use the RECEIVE command to retrieve it. The source account for the file is *SYSTEM*, the user ID is *AUDITS*, and the user class is #SAUDIT. Information Exchange does not prepare a common data header (CDH) to accompany retrieved audit records.

Audits are not available immediately; you cannot successfully issue the RECEIVE command after the AUDIT command to receive the audit data. Audits are normally available during the next session. For more information on the AUDIT and RECEIVE commands, see “AUDIT command” on page 80, and “RECEIVE command” on page 106.

The following steps show how you request and receive audit trails and what an audit trail looks like.

1. The following example shows the command file, INMSG, you would use to create an audit file with information about all files received between July 1 and July 5 with a user class of ORDERS.

   ```
   AUDIT STARTDATE(040701) ENDDATE(040705) CLASS(ORDERS);
   ```

   The response file, OUTMSG, indicates a RETURN(00000) if the audit command was processed properly. The result would be a file placed in your mailbox with account *SYSTEM* and user ID *AUDITS*, with user class #SAUDIT.

2. The next step is to receive the audit file from your mailbox.

   The following example shows two commands, either of which could be used in the command file, INMSG, to receive the audit files.

   ```
   RECEIVE FILEID(AUDITS.FILE) CLASS(#SAUDIT);
   RECEIVE FILEID(AUDITS.FILE) ACCOUNT(*SYSTEM*) USERID(*AUDITS*);
   ```

   If there was an audit file in your Information Exchange mailbox, the response file, OUTMSG, would have a RECEIVED record indicating the file was received.

3. The following example shows the audit trail as it would appear in the file called audits.file.

   NOTE: When the audit trail is received from your mailbox, the data will be divided into records based on the record length of the file to which it is written. For this example, the audit record has been formatted in order to display it on the page.

   ```
   ACCT   SENDER    1    ACCT    USERA   A6943887.554CAE3F51RECEIVED
   SORDERS              EXP/OS2 410 0000000200053473   040702123630
   0000000 040702143701040702123703040702123704
   ```

See the *Information Exchange Interface Programming Guide* for the format of an audit trail record.

Creating distribution lists

You can send the same file to more than one person at a time by making a list of users and sending the file to that list. There are two types of distribution lists: permanent and temporary.

Permanent distribution lists are permanently stored in Information Exchange. The advantage of this type of list is that many people using different types of computers can use it. To create a permanent list, use the LIST command or Information Exchange Administration Services.

Temporary distribution lists last only for the duration of your Information Exchange session. When your Information Exchange session ends, Expedite Base/MVS deletes your temporary list. To create a temporary list, use the LIST command. For a detailed description of the LIST command, see “LIST command” on page 93.

ATTENTION: If you use the LIST command with session-level recovery, do not use the same distribution list name in more than one LIST command in the same session. If you do, Information Exchange replaces the existing list definitions with the new list definitions. Therefore, any file you send uses the most recent list that you define.

Free-format messages

Some Information Exchange interfaces support a free-format message facility. With this facility, you can type short messages on panels and send them to other users. These messages are made up of 79-byte fixed-length records and are sent in message class FFMSG001.

Expedite Base/MVS does not provide free-format message panels. However, you can create free-format messages using an editor and send them using the SEND command.

You can also receive free-format messages with the RECEIVE command. Do this by using the FORMAT parameter of these commands.

Use the FORMAT(y) parameter of the SEND command, and Expedite Base/MVS attempts to send the file as a free-format message. It pads records that are shorter than 79 bytes with blanks and splits records that are longer than 79 bytes into 79-byte records. It also sends the file in class FFMSG001 if you do not specify a class with the SEND command. Use the FORMAT(y) parameter of the RECEIVE command, and Expedite Base/MVS formats the file as a free-format message.

Validating addresses, payment levels, and authorizations when sending data

To reduce transmission costs and ensure that your message can be delivered when communicating with unfamiliar trading partners, you may want to verify the following:

- The address to which you want to send the message is a valid address.
- You can use a particular payment level when sending the message to the intended destinations.
- The intended destinations have appropriate authorization levels to receive messages from your Information Exchange address.

Expedite Base/MVS can check for these items before sending a file, provided you are on the same Information Exchange system as the intended destinations. You can ask Expedite Base/MVS to do this by using the VERIFY parameter of the SEND, SENDEDI, and SENDSTREAM commands. You are normally not charged to use the VERIFY parameter. However, in some circumstances you may incur a charge:

If the destination system is:	And the destination:	Then a charge is:
On the same system	Can be verified	Not incurred

On the same system	Is invalid	Incurred
Another Information Exchange system	Cannot be verified	Not incurred

You can use the VERIFY parameter for distribution lists and individual destinations. However, when used to verify a distribution list, this parameter only verifies that the distribution list exists, not that the individual entries in the list are valid.

If a verification fails, Expedite Base/MVS does not send the data. You can use the VERIFY parameter to specify whether or not Expedite Base/MVS should send the data even if Information Exchange cannot verify the destination of a trading partner on another system.

To indicate how you, as the sender, want to pay the file charges, use the CHARGE parameter of the SEND, SENDEDI, and SENDSTREAM commands. You can specify one of six payment options. For more information, refer to the *Information Exchange Charges Reference.*

For more information on the VERIFY and CHARGE parameters, see “SEND command” on page 118, ““SENDEDI command” on page 124 and “SENDSTREAM command” on page 129.

Using acknowledgments

You can request three types of acknowledgments by using the ACK parameter on any of the Expedite Base/MVS send commands:

receipt	Information Exchange generates a receipt acknowledgment when a message reaches the receiver’s mailbox after a successful Expedite Base/MVS session.
delivery	Information Exchange generates a delivery acknowledgment when a destination user receives a message from the Information Exchange mailbox.
purge	Information Exchange generates a purge acknowledgment when a message is purged from the receiver’s mailbox.

The following table shows the format of an acknowledgment.

Field number	Column	Size	Name	Description
1	1	5	MSGNUMBR	Message number
2	6	2	MSGSEVCD	Severity code
3	8	3	CMDLNTH	Length of command responsible
4	11	5	MSGDESLN	Length of message text
5	16	x	MSGRESPN	Command responsible
6	16 + x	y	MSGDESCR	Message text

The **x** in field 5 represents the length specified by the CMDLNTH field, and the **y** in field 6 represents the length specified by the MSGDESLN field.

MSGNUMBR
The message number for acknowledgments:

01000	Delivery acknowledgment.

01001	Receipt acknowledgment.
01002	Purge acknowledgment.

MSGSEVCD
The severity code for acknowledgments, which is always 00.

CMDLNTH
The length of the MSGRESPN (command responsible) field.

MSGDESLEN
The length of the MSGDESCR (Information Exchange message text) field.

MSGRESPN
The command responsible for the generation of the message, which for all acknowledgments is the send message command. Expedite Base/MVS generates the message in the Information Exchange format for the SEND, SENDSTREAM, SENDEDI, or PUTMEMBER command.

MSGDESCR
The message text generated by Information Exchange. For each type of acknowledgment, the text generated is:

receipt
: n GROUP ACCEPTED FOR y z

  n is the UNIQUEID assigned by Expedite Base/MVS on the SEND, SENDSTREAM, SENDEDI, or PUTMEMBER command. You can use this value to associate the acknowledgment with a file you sent.

  y is replaced with the word LIST or USER, depending on whether the destination was a named distribution list or a single Information Exchange account and user ID.

  z is the account and user ID, list name, or alias and alias name to which the file was delivered.

delivery
: n GROUP DELIVERED TO USER z.

  n is the UNIQUEID assigned by Expedite Base/MVS on the SEND, SENDSTREAM, SENDEDI, or PUTMEMBER command. You can use this value to associate the acknowledgment with a file you sent.

  z is the account and user ID or the alias and alias name to which the file was delivered.

purge
: n GROUP PURGED FROM THE QUEUE OF y. Reason = z

  n is the UNIQUEID assigned by Expedite Base/MVS on the SEND, SENDSTREAM, SENDEDI, or PUTMEMBER command. You can use this value to associate the acknowledgment with a file you sent.

  y is the account and user ID or the alias and alias name of the user from whose mailbox the file was purged.

  z is a simple explanation of why the message was purged.

Working with libraries

Expedite Base/MVS provides four commands that allow you to work with libraries.

The GETMEMBER command allows you to copy a library member from an existing Information Exchange library to an Information Exchange mailbox. The PUTMEMBER command allows you to add a library member to an existing Information Exchange library. There is a charge for using the GETMEMBER and PUTMEMBER commands.

Expedite Base/MVS also provides two commands, LISTLIBRARIES and LISTMEMBERS, which allow you to search for libraries and members to which you have access.

To use libraries in Expedite Base/MVS:

1. Create the library using Information Exchange Administration Services If you do not have access to Information Exchange Administration Services, contact your marketing representative or service administrator.

2. Use the PUTMEMBER and GETMEMBER commands to put members into the library, retrieve them from the library, and place them into your mailbox.

   For more information on these commands, see “GETMEMBER command” on page 90 and “PUTMEMBER command” on page 102.

3. Use the LISTLIBRARIES and LISTMEMBERS commands to identify libraries and members to which you have access.

   For more information on these commands, see “LISTLIBRARIES command” on page 95 and “LISTMEMBERS command” on page 96.

Data compression

If you have installed the appropriate compression software, Expedite Base/MVS will compress and decompress the data you send and receive. To use this facility, you must specify a valid COMPRESS parameter on the SEND, SENDEDI, or SENDSTREAM commands. For more information, see Appendix E, “Using data compression.”

Chapter 9

Expedite Base/MVS response records

A response file consists of echoed commands that Expedite Base/MVS has processed, followed by response records associated with those commands. Expedite Base/MVS provides you with two primary response files, referenced by ddnames OUTPRO and OUTMSG, that correspond to the two command files referenced by ddnames INPRO and INMSG. If you are using checkpoint-level recovery, OUTWORK contains commands and response records processed since the last checkpoint.

OUTPRO response records

Response records contained in the OUTPRO file include:

- PROFILERC, page 144
  Indicates the completion of an entire INPRO file.
- RETURN, page 145
  Indicates the completion of a command in INPRO.
- WARNING, page 146
  Indicates a minor problem that did not stop the command from finishing, but that should be noted.

Detailed information on the records listed above is provided on the following pages.

PROFILERC record

The PROFILERC record is the last record in OUTPRO. The PROFILERC record indicates the profile commands have completed processing. A zero value indicates that all the profile commands to build or alter the profile completed normally.

Format

```
PROFILERC(return code) ERRDESC(error description);
```

NOTE: There is only one PROFILERC record in each OUTPRO.

Parameters

PROFILERC
Indicates the success of the profile commands processing. If the return code is zero, the processing completed normally. If the return code is not zero, Expedite Base/MVS displays an error code, and you may see an ERRDESC value. Expedite Base/MVS displays the same return code on the SESSIONEND record in the OUTMSG file. This parameter contains 5 numeric characters.

ERRDESC
A short description of the error condition. If the return code is zero, Expedite Base/MVS does not display this parameter. This parameter contains 1 to 76 alphanumeric characters.

RETURN record

The RETURN record indicates the completion of a command in INPRO or INMSG. A zero value indicates that the command completed normally.

Format

```
RETURN(return) ERRDESC(error description)
REASON(reason) ERRTEXT(error text);
```

Parameters

RETURN
Indicates completion of the Expedite Base/MVS command. If the return code is zero, the command completed normally. If the return code is not zero, Expedite Base/MVS displays an error number and may also display an ERRDESC, REASON, and ERRTEXT value. This parameter contains 5 numeric characters.

ERRDESC
A short description of the error condition. If the return code is zero, Expedite Base/MVS does not display this parameter. This parameter contains 1 to 76 alphanumeric characters.

REASON
If Expedite Base/MVS includes REASON in the record, it gives more information about what caused the error. Expedite Base/MVS usually includes REASON when an error occurs opening a file. This parameter contains 1 to 240 alphanumeric characters.

ERRTEXT
ERRTEXT gives a more detailed description of an error and may suggest steps to correct the problem. Expedite Base/MVS may include this parameter more than once in a record if the error text consists of multiple lines. This parameter contains 1 to 76 alphanumeric characters.

WARNING record

The WARNING record indicates a low-severity problem that does not prevent the profile command from completing.

Format

```
WARNING(warning) ERRDESC(error description);
```

Parameters

WARNING
Indicates the warning code. This parameter contains 5 numeric characters.

ERRDESC
Indicates a description of the problem. This parameter contains 1 to 76 alphanumeric characters.

OUTMSG response records

Response records contained in the OUTMSG file include:

- AUTOEND, page 149
  Indicates that an Information Exchange session ended automatically.
- AUTOSTART, page 150
  Indicates that an Information Exchange session started automatically.
- AVAILABLE, page 151
  Provides information about messages in your Information Exchange mailbox in response to a QUERY command.
- LIBRARYLIST, page 155
  Provides the results of the LISTLIBRARIES command.
- MEMBERLIST, page 157
  Provides the results of the LISTMEMBERS command.
- MEMBERPUT, page 159
  Returns information about the member placed in the library.
- MOVED, page 160
  Tells you how many files Information Exchange moved from archive in response to an ARCHIVEMOVE command.
- MSGRESP, page 161
  Contains information summarizing your Information Exchange mailbox contents. This record is produced in response to the MSGINFO command.
- NOTSENT, page 163
  Provides information on the data that could not be sent when a destination verification failure occurs with the SENDEDI command.
- RECEIVED, page 165
  Provides information on the data received with a RECEIVE, RECEIVEEDI, or RECEIVESTREAM command.
- RETURN, page 171
  Indicates the completion of a command in INMSG.
- SENT, page 172
  Provides information on the data sent with a SEND, SENDEDI, or SENDSTREAM command.
- SESSIONEND, page 174
  Indicates the return code for the last time you ran Expedite Base/MVS.
- SESSIONRESP, page 175
  Contains information summarizing the current Information Exchange session. This record is produced in response to the SESSIONINFO command.

- STARTED, page 178
  Contains information about the current Information Exchange session and the prior Information Exchange session.
- WARNING, page 180
  Indicates a minor problem that did not stop the command from finishing, but that should be noted.

Detailed information on the records listed above is available on the following pages.

AUTOEND record

The AUTOEND record indicates that Expedite Base/MVS automatically ended an Information Exchange session.

Format

```
AUTOEND;
```

AUTOSTART record

The AUTOSTART record indicates that Expedite Base/MVS started an Information Exchange session automatically.

Format

```
AUTOSTART SESSIONKEY(session key);
```

Parameter

SESSIONKEY
Session access key provided by Information Exchange upon a successful session start. If the session start is not successful, Expedite Base/MVS does not put the session key in the record.

AVAILABLE record

Expedite Base/MVS produces an AVAILABLE record for every message in the mailbox when you use the QUERY command. Each record contains information describing the message. Note that because parameters with blank values do not print, and because some parameters come from the CDH, all of the parameters indicated below may not be included with every AVAILABLE record. If you specify CDH(y) for a message or file that has no common data header, the CDH parameters adopt default values.

Format

```
AVAILABLE

SYSID(system ID) ACCOUNT(account) USERID(user ID)
   or
ACCOUNT(account) USERID(user ID)

MSGKEY(message key) CLASS(class) MODE(t) LENGTH(length) MSGDATE(message
date)
MSGDATELONG(message date long format)
MSGTIME(message time) MSGNAME(message name) MSGSEQNO(message sequence
number)
SYSNAME(system name) SYSLEVEL(system level)
DATATYPE(E|B) EDITYPE(EDI type)
SENDERFILE(sender file) SENDERLOC(sender location) FILEDATE(file date)
FILEDATELONG(file date long format)
FILETIME(file time) RECFM(record format) RECLEN(record length)
RECDLM(C|E|L|N)
DESCRIPTION(description) UNIQUEID(unique ID) CODEPAGE(code page)
SYSTYPE(01|10|11|12|14|15|16|17|19|2|21|22|30|31|33|40|44|71|80|90|91)
SYSVER(system version) TRANSLATE(xlate table)
COMSW(compression software name) COMVER(compression software version)
COMFILE(name of compressed file);
```

NOTE: The following parameters are not shown if you specify CDH(n) using the QUERY command: DATATYPE, SENDERFILE, FILEDATE, FILEDATELONG, RECFM, RECDLM, UNIQUEID, SYSTYPE, TRANSLATE, EDITYPE, SENDERLOC, FILETIME, RECLEN, DESCRIPTION, CODEPAGE, SYSVER, COMSW, COMVER, and COMFILE.

Parameters

SYSID
System ID of the user who sent the message. This parameter contains 1 to 3 alphanumeric characters.

ACCOUNT
Account name of the user who sent the message. This parameter contains 1 to 8 alphanumeric characters.

USERID
User ID of the user who sent the message. This parameter contains 1 to 8 alphanumeric characters.

MSGKEY
A unique identifier assigned to the message or file by Information Exchange. You can use this value for the MSGKEY parameter of the RECEIVE, RECEIVEEDI, or RECEIVESTREAM commands to receive only a specific message or file. This parameter contains 20 hexadecimal characters.

CLASS
User class of the data, specified by the sender. This parameter contains 1 to 8 alphanumeric characters.

MODE
Indicates the network data class field for this data. The sender specified this value. If the file or message is not test mode (indicated by the value T), this parameter is omitted from the record.

LENGTH
Length of the data in the Information Exchange mailbox. The length of the file or message received may be different from the length of the file sent because of reformatting. This parameter contains 1 to 10 numeric characters.

MSGDATE
Date the message was placed into Information Exchange. The format is YYMMDD.

MSGDATELONG
Indicates the long format of MSGDATE. The format is YYYYMMDD. The YY value from MSGDATE is interpreted by Expedite Base/MVS as YYYY.

MSGTIME
Time the message was placed into Information Exchange. The format is HHMMSS.

MSGNAME
Name of the message, specified by the sender. This parameter contains 1 to 8 alphanumeric characters.

MSGSEQNO
Sequence number assigned by the sender to specify a message control number for this data. This parameter contains 1 to 5 alphanumeric characters.

SYSNAME
The name of the system that sent the data. This parameter contains 1 to 8 alphanumeric characters.

SYSLEVEL
The level of the system that sent the data. This parameter contains 1 to 4 alphanumeric characters.

DATATYPE
Indicates whether the data is EBCDIC or binary.

E	EBCDIC
B	Binary

EDITYPE
Type of EDI data available: X12, UCS, UN/TDI, EDIFACT, or UNFORMATTED. This parameter contains 3 to 11 alphanumeric characters.

SENDERFILE
File name the data had on the sender’s system. This parameter contains 1 to 54 alphanumeric characters.

SENDERLOC
Location of the file on the sender’s system. This parameter contains 1 to 65 alphanumeric characters.

FILEDATE
Date of the file on the sender’s system. The format is *YYMMDD*.

FILEDATELONG
Indicates the long format of FILEDATE. The format is *YYYYMMDD*. The *YY* value from FILEDATE is interpreted by Expedite Base/MVS as *YYYY.*

FILETIME
Time of the file on the sender's system. The format is *HHMMSS*.

RECFM
Record format of the file on the sender’s system. If the record format is not appropriate for the sending machine (for example, if the sending machine is a PC), the value is ???? (4 question marks). This parameter contains 1 to 4 alphanumeric characters.

RECLEN
Record length of the file on the sender’s system. This parameter contains 1 to 5 numeric characters.

RECDLM
Indicates what method is used to delimit the records.

C	CRLF characters delimit the records.
E	EDI characters delimit the records. 2-byte length precedes each record.
N	Either the records have no delimiters, or the CDH does not indicate the type of delimiter.

DESCRIPTION
Free-format description of the data given by the sender. This parameter contains 1 to 79 alphanumeric characters.

UNIQUEID
Random ID assigned to the message by the sending interface. It can help you identify the message and also help you match acknowledgments to the message sent. This parameter contains 8 alphanumeric characters.

CODEPAGE
Code page used by the sending system to determine the character representation of the data. This parameter contains 3 numeric characters.

SYSTYPE
Type of system that sent the data. The codes for the systems are:

01	Unknown system type
10	expEDIte/PC
11	Expedite Base/2

12	Expedite Base/AIX
14	Expedite Base for SCO UNIX
15	Expedite Base/DOS
16	Expedite Base for SCO XENIX
17	Expedite Base for Windows
19	Expedite for Windows
20	expEDIte/MVS Host
21	Expedite Base/MVS
22	TCP/IP FTP Gateway
30	Mail Exchange
31	Expedite Base/VM
33	X.400 Gateway
40	Expedite/Direct
44	EDI VAN Interconnect
60	MQSeries Services
61	EDI Services
71	Expedite Base/400
80	Expedite/CICS
90	Information Exchange Administration Services
91	Expedite/Async

This parameter contains 2 hexadecimal digits.

SYSVER
Software version of the system sending the data. This parameter contains 1 numeric character.

TRANSLATE
ASCII-to-EBCDIC translate table used to send this file to Information Exchange. This parameter contains 1 to 8 alphanumeric characters.

COMSW
Name of the compression software package used to compress the file. This parameter contains 10 alphanumeric characters.

COMVER
Version of the compression software package used to compress the file. This parameter contains 1 to 5 alphanumeric characters.

COMFILE
Name of the compressed file. This parameter contains 1 to 54 alphanumeric characters.

LIBRARYLIST record

The LIBRARYLIST record returns information requested by the LISTLIBRARIES command. Each record contains information specific to that library. Parameters with blank values are not included.

Format

```
LIBRARYLIST

OWNER(library owning account) LIBRARY(library name)
MEMBERS(number of members) DESCRIPTION(description)
OWNUSERID(owner's user ID)
CREATEDATE(creation date)
CREATEDATELONG(creation date long format)
CREATETIME(creation time)
UPDATEDBY(ID of last user)
UPDATEDATE(date of last update)
UPDATEDATELONG(date of last update long format)
UPDATETIME(time of last update)
WRITEAUTH(P|O|G|L)
WRITELIST(list of users)
READAUTH(P|O|G|L)
READLIST(distribution list)
SEARCHABLE(Y|N)
OWNERPAYS(Y|N);
```

Parameters

OWNER
Account that owns the library. This parameter contains 1 to 8 alphanumeric characters.

LIBRARY
Name of the library. This parameter contains 1 to 8 alphanumeric characters.

MEMBERS
Number of members present in the library.

DESCRIPTION
Free-format description of the library. This parameter contains 1 to 79 alphanumeric characters.

OWNUSERID
Indicates the user ID of the owner of the library.

CREATEDATE
Indicates the date the library was created. The format is YYMMDD. Information Exchange adjusts the date to that of your local time zone.

CREATEDATELONG
Indicates the long format of CREATEDATE. The format is YYYYMMDD. The YY value from CREATEDATE is interpreted by Expedite Base/MVS as YYYY.

CREATETIME
Indicates the time the library was created. The format is HHMMSS. Information Exchange adjusts the time to that of your local time zone.

UPDATEDBY
Indicates the account ID and user ID, separated by at least one blank, of the user who last redefined this library.

UPDATEDATE
Indicates the date the library was last redefined. The format is YYMMDD. Information Exchange corrects the date to that of your local time zone.

UPDATEDATELONG
Indicates the long format of UPDATEDATE. The format is YYYYMMDD. The YY value from UPDATEDATE is interpreted by Expedite Base/MVS as YYYY.

UPDATETIME
Indicates the time the library was last redefined. The format is HHMMSS. Information Exchange adjusts the time to that of your local time zone.

WRITEAUTH
Indicates the authority type for update access to the library.

P	Only the owner can update this library.
O	Only users with the same account can update this library.
G	Any user can update this library.
L	Any user in the list named in the WRITELIST parameter can update this library.

WRITELIST
Indicates the name of a permanent distribution list that details the users who can update this library.

READAUTH
Indicates the authority type for read access to the library.

P	Only the owner can read the library.
O	Only users with the same account can read this library.
G	Any user can read this library.
L	Any user in the list named in the READLIST parameter can read this library.

READLIST
Indicates the name of a permanent distribution list that details the users who can read this library.

SEARCHABLE
Contains Y if the library is searchable and N if it is not searchable.

OWNERPAYS
Indicates whether the owner of the library is responsible for charges associated with transferring the library member from the user's mail box to the user's system when the library member is retrieved. These are usually called *receive-side* charges.

Y	The owner of the library pays the receive-side charges.
N	The receive-side charges will be charged to you.

No LIBRARYLIST or other response records are written to the message response file if no libraries are found that match the criteria you specified.

MEMBERLIST record

The MEMBERLIST record returns information requested by the LISTMEMBERS command. Each record contains information specific to that member. Parameters with blank values are not included.

Format

```
MEMBERLIST

MEMBER(member name) DESCRIPTION(description)
CREATEDBY(user's ID)
CREATEDATE(creation date)
CREATEDATELONG(creation date long format)
CREATETIME(creation time)
UPDATEDBY(member ID)
UPDATEDATE(date of last update)
UPDATEDATELONG(date of last update long format)
UPDATETIME(time of last update)
LENGTH(member length);
```

Parameters

MEMBER
Name of a library member. This parameter contains 1 to 8 alphanumeric characters.

DESCRIPTION
Free-format description of the member. This parameter contains 1 to 79 alphanumeric characters.

CREATEDBY
Indicates, in a fixed format, the system ID, account ID, and user ID of the user that created this library member. The first 4 characters are the system ID (and may be blank), the next 8 characters are the account ID, and the last 8 characters are the user ID.

CREATEDATE
Indicates the date the member was created. The format is YYMMDD. Information Exchange adjusts the date to that of your local time zone.

CREATEDATELONG
Indicates the long format of CREATEDATE. The format is YYYYMMDD. The YY value from CREATEDATE is interpreted by Expedite Base/MVS as YYYY.

CREATETIME
Indicates the time the member was created. The format is HHMMSS. Information Exchange adjusts the time to that of your local time zone.

UPDATEDBY
Indicates, in a fixed format, the system ID, account ID, and user ID of the user that last updated this library member. The first 4 characters are the system ID (and may be blank), the next 8 characters are the account ID, and the last 8 characters are the user ID.

UPDATEDATE
Indicates the date the member was last updated. The format is YYMMDD. Information Exchange adjusts the date to that of your local time zone.

UPDATEDATELONG
Indicates the long format of UPDATEDATE. The format is *YYYYMMDD*. The *YY* value from UPDATEDATE is interpreted by Expedite Base/MVS as *YYYY.*

UPDATETIME
Indicates the time the member was last updated. The format is *HHMMSS*. Information Exchange adjusts the time to that of your local time zone.

LENGTH
Indicates the length of the member in bytes.

No MEMBERLIST or other response records are written to the message response file if the user has no authority to access the library.

MEMBERPUT record

The MEMBERPUT record returns information about the member placed in the library. Since parameters with blank values are not written, all of the parameters indicated below may not appear with the MEMBERPUT record.

Format

```
MEMBERPUT

OWNER(owning account) LIBRARY(library name) MEMBER(member name)
UNIQUEID(unique ID) LENGTH(file length);
```

Parameters

OWNER
Account that owns the library. This parameter contains 1 to 8 alphanumeric characters.

LIBRARY
Name of the library. This parameter contains 1 to 8 alphanumeric characters.

MEMBER
Name of the member.

UNIQUEID
Random ID assigned to the member by Expedite Base/MVS. It helps to identify the member. This parameter contains 8 alphanumeric characters.

LENGTH
Length of the file sent. This parameter contains 1 to 8 numeric characters.

MOVED record

The MOVED record tells how many files Information Exchange moved from short-term archive to your Information Exchange mailbox as a result of an ARCHIVEMOVE command.

Format

```
MOVED
```

```
NUMBER(number of files moved);
```

Parameter

NUMBER
Number of files moved from short-term archive to your Information Exchange mailbox as a result of an ARCHIVEMOVE command. If this value is zero, no files in the archive match the ARCHIVEID specified on the ARCHIVEMOVE command, or the files have already been retrieved and still reside in your mailbox. This parameter contains 1 to 5 numeric characters.

MSGRESP record

Expedite Base/MVS produces a MSGRESP record in response to the MSGINFO command. It contains information summarizing your Information Exchange mailbox contents.

Format

```
MSGRESP

DATE(date) DATELONG(date long format) TIME(time)
DATAMSGS(data messages) DATADATE(data date)
DATADATELONG(data date long format)
DATATIME(data time) SERVMSGS(service messages)
SERVDATE(service date) SERVDATELONG(service date long format)
SERVTIME(service time) TOTALDATA(total data) MAXMSG(maximum message);
```

Parameters

DATE
Date that Information Exchange generated the summary information. The format is *YYMMDD*.

DATELONG
Indicates the long format of DATE. The format is *YYYYMMDD*. The *YY* value from DATE is interpreted by Expedite Base/MVS as *YYYY.*

TIME
Time that Information Exchange generated the summary information. The format is *HHMMSS*.

DATAMSGS
Number of Information Exchange segments available as actual data. This parameter contains 1 to 6 characters.

DATADATE
Date of oldest data message available. The format is *YYMMDD*.

DATADATELONG
Indicates the long format of DATADATE. The format is *YYYYMMDD*. The *YY* value from DATADATE is interpreted by Expedite Base/MVS as *YYYY.*

DATATIME
Time of the oldest data message available. The format is *HHMMSS*.

SERVMSGS
Number of Information Exchange segments available as service messages from Information Exchange. This parameter contains 1 to 6 numeric characters.

SERVDATE
Date of the oldest Information Exchange service message available. The format is *YYMMDD*.

SERVDATELONG
Indicates the long format of SERVDATE. The format is *YYYYMMDD*. The *YY* value from SERVDATE is interpreted by Expedite Base/MVS as *YYYY.*

SERVTIME
Time of the oldest Information Exchange service message available. The format is *HHMMSS*.

TOTALDATA
Total number of Information Exchange segments available. It includes both service messages from Information Exchange and actual data messages. This parameter contains 1 to 6 numeric characters.

MAXMSG
Total number of Information Exchange segments contained in the largest Information Exchange message or message group in the mailbox. This parameter contains 1 to 6 numeric characters.

NOTSENT record

Expedite Base/MVS produces a NOTSENT record for every EDI envelope that could not be sent due to a destination verification failure. NOTSENT records are produced only when the VERIFY parameter of the SENDEDI command is set to c or g.

Because parameters with blank values do not print, Expedite Base/MVS might not include all the parameters listed below with each NOTSENT record.

Format

```
NOTSENT

ALIAS(alias) ALIASNAME(alias name)
   or
SYSID(system ID) ACCOUNT(account) USERID(user ID)
   or
ACCOUNT(account) USERID(user ID)
   or
LISTNAME(list name)

EDITYPE(datatype) DESTINATION(destination) QUALIFIER(qualifier)
CONTROLNUM(control number) CLASS(class) MSGNAME(message name)
MSGSEQNO(message sequence number);
```

Parameters

ALIAS
Alias table type and table name of the Information Exchange destination.

G*xxx*	Global alias table, where *xxx* identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where *xxx* identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where *xxx* identifies a 1 to 3-character table name.

This parameter contains 1 to 4 characters.

ALIASNAME
Alias name defined in the alias table. This parameter contains 1 to 16 alphanumeric characters.

SYSID
System ID of the Information Exchange destination. This parameter contains 1 to 3 alphanumeric characters.

ACCOUNT
Account of the Information Exchange destination. This parameter contains 1 to 8 alphanumeric characters.

USERID
User ID of the Information Exchange destination. This parameter contains 1 to 8 alphanumeric characters.

LISTNAME
Name of a previously defined list of account and user IDs used as the Information Exchange destination. This parameter contains 1 to 8 alphanumeric characters.

EDITYPE
Value that indicates the type of EDI data that would have been sent, had the destination verification failure not occurred: X12, UCS, UN/TDI, or EDIFACT. This parameter contains 1 to 7 alphanumeric characters.

DESTINATION
Destination specified in the EDI data. This parameter contains 1 to 35 alphanumeric characters.

QUALIFIER
EDI qualifier for the destination. This parameter contains 1 to 4 alphanumeric characters.

CONTROLNUM
Interchange Control Number from the X12 or UCS data. For UN/TDI data, this is the SNRF element. For EDIFACT data, it is the data element 0020 (Interchange Control Reference). This parameter contains 1 to 14 alphanumeric characters.

CLASS
User class obtained from the EDI data, the CLASS parameter, or the default. This parameter contains 1 to 8 alphanumeric characters.

MSGNAME
Message name obtained from the EDI data or the MSGNAME parameter. This parameter contains 1 to 8 alphanumeric characters.

MSGSEQNO
Message sequence number obtained from the MSGSEQNO parameter or generated by Expedite Base/MVS. This parameter contains 1 to 5 alphanumeric characters.

RECEIVED record

Expedite Base/MVS produces a RECEIVED record for every file or EDI envelope received from Information Exchange. This record contains information describing the received data. Because parameters with blank values do not print, and because the sender may not have provided some parameters, Expedite Base/MVS may not include all of the parameters listed below in every RECEIVED record.

Format

```
RECEIVED

ALIAS(alias) ALIASNAME(alias name)
   or
SYSID(system ID) ACCOUNT(account) USERID(user ID)
   or
ACCOUNT(account) USERID(user ID)
   or
RECEIVER(receiver ID) RECVQUAL(receiver qualifier)

SENDER(sender ID)
SENDQUAL(sender qualifier) CONTROLNUM(control number)
CLASS(class) MODE(T) PRIORITY(A|P) CHARGE(1|5|6) ACK(D)
LENGTH(file length) FILEID(file ID) MSGDATE(message date)
MSGDATELONG(message date long format) MSGTIME(message time)
MSGSEQO(IE message number) MSGNAME(message name) MSGSEQNO(msesage
sequence number)
SESSIONKEY(session key) DELIMITED(C|L|E|N)
SYSNAME(system name) SYSLEVEL(system level)
STARTDATE(starting date) STARTTIME(starting time)
ENDDATE(ending date) ENDTIME(ending time) TIMEZONE(L|G)DATATYPE(E|B)
EDITYPE(EDI type)
SENDERFILE(sender file) SENDERLOC(sender location)FILEDATE(file date)
FILEDATELONG(file date long format) FILETIME(file time)
RECFM(record format) RECLEN(record length) RECDLM (C|E|L|N)
DESCRIPTION(description) UNIQUEID(unique ID)
SYSTYPE(01|10|11|12|14|15|16|17|19|20|21|22|30|31|33|40|44|71|80|90|91)
CODEPAGE(code page) SYSVER(system version) TRANSLATE(xlate table)
COMSW(compression software name) COMVER(compression software version)
COMFILE(name of compressed file) DCMPRC(decompression return code);
```

Parameters

ALIAS
Indicates the table type and table name of an alias table.

G*xxx*	Global alias table, where *xxx* identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where *xxx* identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where *xxx* identifies a 1 to 3-character table name.

This parameter contains 1 to 4 alphanumeric characters.

ALIASNAME
Alias name defined in the alias table. This parameter contains 1 to 16 alphanumeric characters.

SYSID
System ID of the user who sent the message. This parameter contains 1 to 3 alphanumeric characters.

ACCOUNT
Account of the user who sent the message. This parameter contains 1 to 8 alphanumeric characters.

USERID
User ID of the user who sent the message. This parameter contains 1 to 8 alphanumeric characters.

RECEIVER
Receiver ID specified in the EDI data. This parameter contains 1 to 35 characters.

RECVQUAL
EDI qualifier for the receiver specified in the EDI data. This parameter contains 1 to 4 characters.

SENDER
Sender ID specified in the EDI data. This parameter contains 1 to 35 characters.

SENDQUAL
EDI qualifier for the sender specified in the EDI data.

CONTROLNUM
Interchange Control Number from the X12 or UCS data. For UN/TDI data, this is the SNRF element. For EDIFACT data, this is the element 0020 (Interchange Control Reference). This parameter contains 1 to 14 characters.

CLASS
User class of the data specified by the sender to identify the data. This parameter contains 1 to 8 alphanumeric characters.

MODE
Sender-specified network data class field for this data. This parameter does not appear in the record for a message that is not test mode (indicated by the value T).

PRIORITY
Class of delivery service for this file. This parameter does not appear in the record for a normal-priority message.

A	Normal-priority re queued archive message
P	High priority

CHARGE
Arrangements for the file charges.

1	Receiver pays all charges.
5	Sender and receiver split the charges.
6	Sender pays all charges.

ACK
If the sender asked Information Exchange to send a delivery acknowledgment, this value is set to D. Otherwise, this parameter is omitted.

LENGTH
Length of the received file. This value is cumulative for all files received during the RECEIVE or RECEIVEEDI command. This value begins with zero and is cumulative for the RECEIVES-TREAM command. This parameter contains 1 to 9 numeric characters.

FILEID
Name of the file in which Expedite Base/MVS placed the data received. This parameter contains 1 to 54 alphanumeric characters.

MSGDATE
Date the data received was placed into Information Exchange. The format of this parameter is *YYMMDD*.

MSGDATELONG
Indicates the long format of MSGDATE. The format is *YYYYMMDD*. The *YY* value from MSGDATE is interpreted by Expedite Base/MVS as *YYYY.*

MSGTIME
Time the data received was placed into Information Exchange. The format of this parameter is *HHMMSS*.

MSGSEQO
Unique number assigned to the data by Information Exchange. This parameter contains 1 to 6 numeric characters.

MSGNAME
Name for the data specified by the sender. This parameter contains 1 to 8 alphanumeric characters.

MSGSEQNO
Number assigned by the sender as a file control number for this data. This parameter contains 1 to 5 alphanumeric characters.

SESSIONKEY
The session access key that Expedite Base/MVS used when the file was received. This value is the archive ID for the file if you did not specify an ARCHIVEID parameter in the RECEIVE, RECEIVEEDI, or RECEIVESTREAM command. This parameter contains 1 to 8 alphanumeric characters.

DELIMITED
Indicates the way Expedite Base/MVS actually processed record delimiters when the file was received.

C	Expedite Base/MVS split records at carriage-return/line-feed (CRLF) characters.
L	Expedite Base/MVS split records according to the 2-byte length delimiters at the beginning of each record.
E	Expedite Base/MVS split records according to EDI delimiters. If you specified EDIOPT(n), the records are split according to the length of the data set.
N	Expedite Base/MVS stored the data as it was received. The record length of the data depends on the record length of the data set allocated to receive the data, or on the format option.

SYSNAME
Name of the system that sent the data. This parameter contains 1 to 8 alphanumeric characters.

SYSLEVEL
Level of the system that sent the data. This parameter contains 1 to 4 alphanumeric characters.

STARTDATE
Starting date for files received from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. This parameter contains up to 8 numeric characters.

STARTTIME
Starting time for files received from Information Exchange. The format is *HHMMSS*. This parameter contains 6 numeric characters.

ENDDATE
Ending date for files received from Information Exchange. The format is *YYMMDD* or *YYYYMMDD*. This parameter contains up to 8 numeric characters.

ENDTIME
Ending time for files received from Information Exchange. The format is *HHMMSS*. This parameter contains 6 numeric characters.

TIMEZONE
Time zone reference for the STARTTIME and ENDTIME parameters.

l	Local time, as defined on the TIMEZONE parameter of the IDENTIFY command.
g	Greenwich mean time (GMT)

DATATYPE
Type of data (EBCDIC or binary).

E	EBCDIC
B	Binary

EDITYPE
Type of EDI data received (X12, UCS, UN/TDI, EDIFACT, or UNFORMATTED). This parameter contains 1 to 11 alphanumeric characters.

SENDERFILE
File name the data had on the sender's system. This parameter contains 1 to 54 alphanumeric characters.

SENDERLOC
Location of the file on the sender's system. This parameter contains 1 to 65 alphanumeric characters.

FILEDATE
Date of the file on the sender's system. The format is YYMMDD.

FILEDATELONG
Indicates the long format of FILEDATE. The format is YYYYMMDD. The YY value from FILEDATE is interpreted by Expedite Base/MVS as YYYY.

FILETIME
Time of the file on the sender's system. The format is HHMMSS.

RECFM
Record format of the file on the sender’s system. If the record format is not used by the sending machine (for example, if the sending machine is a PC), the value is ????.

RECLEN
Record length of the file on the sender’s system. This parameter contains 1 to 5 numeric characters.

RECDLM
Method used to delimit the records.

C	CRLF characters delimit the records.
E	EDI characters delimit the records.
L	A 2-byte length preceding each record delimits the records.
N	The records contain no delimiters, or the sender did not indicate the type of delimiters.

DESCRIPTION
Free-format description of the data given by the sender. This parameter contains 1 to 79 alphanumeric characters.

UNIQUEID
Random ID assigned to the data by the sending interface. It helps you identify the data. This parameter contains 8 alphanumeric characters.

CODEPAGE
Code page used by the sending system to determine the character representation of a given byte. This parameter contains 3 numeric characters.

SYSTYPE
Type of system that sent the data. The codes for the systems are:

01	Unknown system type
10	expEDIte/PC
11	Expedite Base/2
12	Expedite Base/AIX
14	Expedite Base for SCO UNIX
15	Expedite Base/DOS
16	Expedite Base for SCO XENIX
17	Expedite Base for Windows
19	Expedite for Windows
20	expEDIte/MVS Host
21	Expedite Base/MVS
22	TCP/IP FTP Gateway
30	Mail Exchange
31	Expedite Base/VM

33	X.400 Gateway
40	Expedite/Direct
44	EDI VAN Interconnect
60	MQSeries Services
61	EDI Services
71	Expedite Base/400
80	Expedite/CICS
90	Information Exchange Administration Services
91	Expedite/Async

SYSVER
Software version of the system sending the data. This parameter contains 1 to 3 numeric characters.

TRANSLATE
ASCII-to-EBCDIC translate table used when this file was sent to Information Exchange. This parameter contains 1 to 8 alphanumeric characters.

COMSW
Name of the compression software package used to compress the file. This parameter contains 10 alphanumeric characters.

COMVER
Version of the compression software package used to compress the file. This parameter contains 1 to 5 alphanumeric characters.

COMFILE
Name of the compressed file. This parameter contains 1 to 54 alphanumeric characters.

DCMPRC
Return code from decompression processing. This parameter contains 1 to 5 alphanumeric characters.

RETURN record

The RETURN record indicates the completion of a command in INPRO or INMSG. A zero value indicates that the command completed normally.

Format

```
RETURN(return) ERRDESC(error description)
REASON(reason) ERRTEXT(error text)...
SESSIONKEY(session key);
```

Parameters

RETURN
Completion code for the Expedite Base/MVS command. If the return code is zero, the command completed normally. If the return code is not zero, Expedite Base/MVS displays an error number and possibly an ERRDESC, REASON, and ERRTEXT value. This parameter contains 5 numeric characters.

ERRDESC
A short description of the error condition. If the return code is zero, Expedite Base/MVS does not display this parameter. This parameter contains 1 to 76 alphanumeric characters.

REASON
If Expedite Base/MVS includes REASON in the record, it gives more information about what caused the error. Expedite Base/MVS usually includes REASON when an error occurs opening a file. This parameter contains 1 to 240 alphanumeric characters.

ERRTEXT
ERRTEXT gives a more detailed description of an error and might suggest steps to correct the problem. Expedite Base/MVS might include this parameter more than once in a record if the error text consists of multiple lines. This parameter contains 1 to 76 alphanumeric characters.

SESSIONKEY
Session access key provided by Information Exchange upon a successful session start. The session access key is only provided after a START command. When the RETURN record contains a session access key, this is the only parameter included in the record.

SENT record

The SENT record returns information about the data sent that might not be apparent from the command parameters. This record appears only in OUTMSG. Expedite Base/MVS creates one SENT record for each file, stream, or EDI envelope transmitted. Because parameters with blank values do not print, Expedite Base/MVS might not include all the parameters listed below with each SENT record.

When the SENT record follows a SEND or SENDSTREAM command, it includes only the UNIQUEID and the LENGTH parameters.

When the SENT record follows a SENDEDI command, it returns all the parameters shown, unless the parameter value is blank.

Format

```
SENT

ALIAS(alias) ALIASNAME(alias name)
   or
SYSID(system ID) ACCOUNT(account) USERID(user ID)
   or
ACCOUNT(account) USERID(user ID)
   or
LISTNAME(list name)

UNIQUEID(unique ID) LENGTH(length) EDITYPE(datatype)
DESTINATION(destination) QUALIFIER(qualifier)
CONTROLNUM(control number) CLASS(class)
MSGNAME(message name) MSGSEQNO(message sequence number);
```

Parameters

ALIAS
Alias table type and table name of the Information Exchange destination.

G*xxx*	Global alias table, where *xxx* identifies a 1 to 3-character table name.
O*xxx*	Organizational alias table, where *xxx* identifies a 1 to 3-character table name.
P*xxx*	Private alias table, where *xxx* identifies a 1 to 3-character table name.

This parameter contains 1 to 4 characters.

ALIASNAME
Alias name defined in the alias table. This parameter contains 1 to 16 alphanumeric characters.

SYSID
System ID of the Information Exchange destination. This parameter contains 1 to 3 alphanumeric characters.

ACCOUNT
Account of the Information Exchange destination. This parameter contains 1 to 8 alphanumeric characters.

USERID
User ID of the Information Exchange destination. This parameter contains 1 to 8 alphanumeric characters.

LISTNAME
Name of a previously defined list of account and user IDs used as the Information Exchange destination. This parameter contains 1 to 8 alphanumeric characters.

UNIQUEID
Random ID assigned to the message by Expedite Base/MVS to help you identify the message. The first 8 characters of the MSGDESCR parameter in the Information Exchange acknowledgment contain the UNIQUEID. See “Information Exchange generated message text” in the *Information Exchange Interface Programming Guide* for more details on the format of the acknowledgment. This parameter contains 8 alphanumeric characters.

LENGTH
Length of the file, stream, or EDI envelope sent. This parameter contains 1 to 9 numeric characters.

EDITYPE
Value that indicates the type of EDI data sent: X12, UCS, UN/TDI, or EDIFACT. This parameter contains 1 to 7 alphanumeric characters.

DESTINATION
Destination specified in the EDI data. This parameter contains 1 to 35 alphanumeric characters.

QUALIFIER
EDI qualifier for the destination. This parameter contains 1 to 4 alphanumeric characters.

CONTROLNUM
Interchange control number from the X12 or UCS data. For UN/TDI data, this is the SNRF element. For EDIFACT data, it is the data element 0020 (Interchange control reference). This parameter contains 1 to 14 alphanumeric characters.

CLASS
User class obtained from the EDI data, the CLASS parameter, or the default. This parameter contains 1 to 8 alphanumeric characters.

MSGNAME
Message name obtained from the EDI data or the MSGNAME parameter. This parameter contains 1 to 8 alphanumeric characters.

MSGSEQNO
Message sequence number obtained from the MSGSEQNO parameter or generated by Expedite Base/MVS. This parameter contains 1 to 5 alphanumeric characters.

SESSIONEND record

The SESSIONEND record is the last record in the response file. The SESSIONEND record indicates the completion of an entire INMSG file. A zero value indicates that all the commands in the file completed normally.

Format

```
SESSIONEND(return code) REASON(reason) ERRDESC(error description)
ERRTEXT(error text);
```

NOTE: There is only one SESSIONEND record per response file, even if there are multiple START and END commands.

Parameters

RETCODE
Completion code for the Expedite Base/MVS command. If the return code is zero, the command completed normally. If the return code is not zero, Expedite Base/MVS displays an error number and possibly an ERRDESC, REASON, and ERRTEXT value. This parameter contains 5 numeric characters.

ERRDESC
A short description of the error condition. If the return code is zero, Expedite Base/MVS does not display this parameter. If the return code is not zero, Expedite Base/MVS might display this parameter. This parameter contains 1 to 76 alphanumeric characters.

REASON
If Expedite Base/MVS includes REASON in the record, it gives more information about what caused the error. Expedite Base/MVS usually includes REASON when an error occurs opening a file. This parameter contains 1 to 240 alphanumeric characters.

ERRTEXT
ERRTEXT gives a more detailed description of an error and might suggest steps to correct the problem. Expedite Base/MVS can include this parameter more than once in a record if the error text consists of multiple lines. This parameter contains 1 to 76 alphanumeric characters.

SESSIONRESP record

Expedite Base/MVS produces a SESSIONRESP record in response to the SESSIONINFO command. It contains information summarizing the current Information Exchange session.

Format

```
SESSIONRESP

DATE(date) DATELONG(date long format) TIME(time) SESSDATE(session date)
SESSDATELONG(session date long format) SESSTIME(session time)
SESSLEN(session length) SEGSENT(segment sent) RCVACTIVE(command
received)
SEGRCVD(segment received) IDLETIME(idle time) SYSWAIT(P|C)
MSGSENT(message sent) MSGOK(message ok) MSGREJECT(message reject)
DATARCVD(data received) SERVRCVD(service received)
RESPRCVD(response received) SENDCKPT(send checkpoint)
RCVCKPT(receive checkpoint) SENDID(send ID) RCVID(receive ID);
```

Parameters

DATE
Date that Information Exchange generated the session information. The format is *YYMMDD*.

DATELONG
Indicates the long format of DATE. The format is *YYYYMMDD*. The *YY* value from DATE is interpreted by Expedite Base/MVS as *YYYY*.

TIME
Time that Information Exchange generated the session information. The format is *HHMMSS*.

SESSDATE
Date the current Information Exchange session started. The format is *YYMMDD*.

SESSDATELONG
Indicates the long format of SESSDATE. The format is *YYYYMMDD*. The *YY* value from SESSDATE is interpreted by Expedite Base/MVS as *YYYY*.

SESSTIME
Time the current Information Exchange session started. The format is *HHMMSS*.

SESSLEN
Length of the current Information Exchange session. The format is *HHMMSS*.

SEGSENT
Number of segments sent to Information Exchange by Expedite Base/MVS between the time the session started and the time the SESSIONINFO command processed. This parameter contains 1 to 6 numeric characters.

RCVACTIVE
Number of Information Exchange RECEIVE commands active at the time the SESSIONINFO command processed. This should be zero under normal conditions. This parameter contains 1 or 2 numeric characters.

SEGRCVD
Number of segments sent to Expedite Base/MVS by Information Exchange between the time the session started and the time the SESSIONINFO command processed. This parameter contains 1 to 6 numeric characters.

IDLETIME
Time elapsed since the last Information Exchange message transfer activity. The format is *HHMMSS*.

SYSWAIT
Indicator of whether Information Exchange is waiting for a commit or pace response from Expedite Base/MVS.

P	Pace response
C	Commit response

Under normal conditions, Expedite Base/MVS does not include this parameter in the SESSIONRESP record.

MSGSENT
Number of messages sent to Information Exchange by Expedite Base/MVS between the time the session started and the time the SESSIONINFO command processed. Because Expedite Base/MVS uses message grouping, this value might not correspond to the number of files sent by Expedite Base/MVS. This parameter contains 1 to 6 numeric characters.

MSGOK
Number of messages sent to Information Exchange by Expedite Base/MVS and accepted for delivery between the time the session started and the time the SESSIONINFO command processed. Because Expedite Base/MVS uses message grouping, this value might not correspond to the number of files sent by Expedite Base/MVS. This parameter contains 1 to 6 numeric characters.

MSGREJECT
Number of messages sent to Information Exchange by Expedite Base/MVS, but rejected between the time the session started and the time the SESSIONINFO command processed. Because Expedite Base/MVS uses message grouping, this value might not correspond to the number of files sent by Expedite Base/MVS. This parameter contains 1 to 6 numeric characters.

DATARCVD
Number of data messages sent to Expedite Base/MVS by Information Exchange between the time the session started and the time the SESSIONINFO command processed. Because Expedite Base/MVS uses message grouping, this value might not correspond to the number of files received. This parameter contains 1 to 6 numeric characters.

SERVRCVD
Number of service messages sent to Expedite Base/MVS by Information Exchange between the time the session started and the time the SESSIONINFO command processed. This parameter contains 1 to 6 numeric characters.

RESPRCVD
Number of command responses sent to Expedite Base/MVS by Information Exchange between the time the session started and the time the SESSIONINFO command processed. This parameter contains 1 to 6 numeric characters.

SENDCKPT
Last send checkpoint number at the time the SESSIONINFO command processed. This parameter contains 1 to 5 numeric characters.

RCVCKPT
Last receive checkpoint number at the time the SESSIONINFO command processed. This parameter contains 1 to 5 numeric characters.

SENDID
ID from the last SEND command. It contains the values specified in the ACCOUNT, USERID, MSGNAME, and MSGSEQNO parameters of the SEND command. This parameter contains 1 to 29 alphanumeric characters.

RCVID
ID from the last message received. This parameter contains 1 to 30 alphanumeric characters.

STARTED record

The STARTED record, a session start response record, provides information about the current Information Exchange session and the prior Information Exchange session. This record is written to the output file as a result of a START or AUTOSTART command.

Format

```
STARTED
LASTSESS(0|1) SESSIONKEY(session key)
IEVERSION(version) IERELEASE(release)
RESPCODE(code):
```

Parameters

LASTSESS
Indicates the status of the previous session.

0	Indicates the last session was successful.
1	Indicates the last session was not successful.

SESSIONKEY
Indicates the unique identifier for this Information Exchange session. This is also used as the archive ID for the files that are archived and for which you did not specify an ARCHIVEID on the RECEIVE command. SESSIONKEY is 8 alphanumeric characters.

IEVERSION
Indicates the version of Information Exchange. Levels of Information Exchange are tracked as V.R where V is the version, a major enhancement in the service, and where R is the release, a minor enhancement. IEVERSION is two digits, padded on the left with zeros.

IERELEASE
Indicates the release of Information Exchange. Levels of Information Exchange are tracked as V.R where V is the version, a major enhancement in the service, and where R is the release, a minor enhancement. IERLEASE is two digits, padded on the left with zeros.

RESPCODE
Indicates the Information Exchange response code for the current session (not the previous session). The response code is interpreted for you by Expedite, so if it is not 0, not 2, and not 8, you will get a SESSIONEND return code from Expedite indicating the problem. RESPCODE is 5 digits, padded on the left with zeros.

Using these new features, you can use Expedite to determine if the previous session was successful or not. After a session fails with 29999, follow these steps:

1. Specify AUTOSTART(N), AUTOEND(N), and RECOVERY(S) on your TRANSMIT command in the Expedite profile.

2. Create an output file containing START and END records; an example follows:

   ```
   START CHECK(Y);
   END;
   ```

   NOTE: Do not specify any other commands in the input file if you specify CHECK(Y) on the session START command.

3. Run Expedite. No data is transferred in the above example, and you are not charged for this inquiry.

4. Examine the output file to check the LASTSESS parameter value on the STARTED record.

LASTSESS(0)	Indicates the previous session was successful. No further recovery is required.
LASTSESS(1)	Indicates the previous session was not successful.

WARNING record

The WARNING record indicates that a low-severity problem occurred which did not prevent the command from completing.

Format

```
WARNING(warning) ERRDESC(error description);
```

Parameters

WARNING
Warning number. This parameter contains 5 numeric characters.

ERRDESC
Description of the problem. This parameter contains 1 to 76 alphanumeric characters.

Expedite Base/MVS installation overview

Expedite Base/MVS is capable of communicating using TCP/IP or SNA LU 6.2 communications. SSL communications are considered part of TCP/IP communications. Detailed instructions on installation are contained in the program directory shipped with the Expedite Base/MVS product. This section contains an overview of the installation procedure. Most of the steps mentioned here must be performed by your system programmer.

Installing the Expedite Base/MVS program

You install Expedite Base/MVS with System Modification Program/Extended (SMP/E). Instructions on how to install the program with SMP/E are contained in the program directory, which is included with the product tape.

SNA LU 6.2 installation instructions

To enable SNA LU 6.2 communications with Expedite Base/MVS, you will need to order a logical unit (LU) name, define the LU name, and make necessary cross-domain resource (CDRSC) definitions. These items are briefly discussed below.

Order the logical unit (LU) name

Expedite Base/MVS communicates with the Information Exchange Common Front End, formerly known as the LU 6.2 relay.

Before you can use Expedite Base/MVS, you must order the LU name that Expedite Base/MVS uses when communicating with the Information Exchange Common Front End. When you order your LU name, specify that it is to go in session with the Information Exchange Common Front End and is to be used for Expedite Base/MVS. In the U.S., the Information Exchange Common Front End is named ibm0rely.

ATTENTION: When migrating from expEDIte/MVS Host to Expedite Base/MVS, retaining the same LU name can cause difficulties. All definitions on your VTAM system and the Network VTAM system must be updated for a LU 6.2 connection. There is a greater chance of error when setting up these definitions when the same LU name is used. Using the same LU name does not allow a period of time for migration to Expedite Base/MVS. Ordering a new LU name is always recommended.

Your marketing representative can help you with your order.

Define the LU name

After you have ordered the LU name for Expedite Base/MVS, you must define the name as a VTAM application on your system. Sample VTAM definitions and LOGMODE entries are included in the Expedite Base/MVS program directory.

You must also include the LU name in the USERLUNAME parameter of the SNACOMM profile command. See “Profile command file (INPRO)” on page 13 and “SNACOMM command” on page 71 for more information on the SNACOMM command.

CDRSC

You must define the Information Exchange Common Front End as an active CDRSC. Information for the definition is included in the program directory.

TCP/IP installation instructions

When you use TCP/IP communications through the network (whether SSL or non-SSL connectivity), the size of your network, other applications you will be accessing through the network, and your hardware configuration are taken into account. Network personnel will work with your networking personnel to allow communications with the Information Exchange Common Front End.

Your Internet Service Provider must provide you with the connection instructions for TCP/IP communications on the Internet using SSL that apply to their services.

NOTE: The IP address used to communicate using SSL with Information Exchange is different than the IP address used to communicate non-SSL.

Chapter 11

Communicating with Information Exchange using SSL

Expedite Base/MVS 4.6 is capable of communicating with Information Exchange using Secure Socket Layer (SSL) protocol over a TCP/IP connection. When using SSL, a digital certificate (which is associated with your Information Exchange account and user ID) provides strong authentication for the Information Exchange session. Using SSL also provides data integrity and encryption mechanisms to protect your data as it goes across the network.

Expedite Base/MVS passes the digital certificate to an SSL Information Exchange component for verification and strong authentication. The digital certificate is actually a X.509 certificate and must be stored on your z/OS system. The TCP/IP address that you use for the SSL Information Exchange component is different than the TCP/IP address that you use for non-SSL communications.

Internally, Expedite Base/MVS uses the System SSL Cryptographic Services Base element of z/OS. System SSL supports both the TLS (Transport Layer Security) and SSL protocols. Throughout this book, SSL describes both SSL and TLS protocols. Expedite Base/MVS and the SSL Information Exchange component negotiate the security features for each session based on the capabilities installed in System SSL and those available in the SSL Information Exchange component. The SSL Information Exchange component is called the Secure Front End.

System SSL can be set up to use either software algorithms or the IBM Integrated Cryptographic Service Facility (ICSF) for hardware cryptographic support. When ICSF is properly installed, hardware encryption is used during communication. Expedite Base/MVS writes information about the cryptographic support used during the session to ddname SYSOUT.

Network connectivity

The primary purpose of adding SSL communications to Expedite Base/MVS is to permit secure communications over the Internet. However, there may be other networks which permit SSL communications with Information Exchange. To determine your network options for communicating SSL with Information Exchange, contact the Help Desk for your region.

Obtaining and managing X.509 certificates

This section explains the registration and setup required to use SSL protocol over TCP/IP with Expedite Base/MVS. Users with existing Information Exchange connectivity can register to obtain their X.509 certificate. All other users should contact their local Help Desk for assistance. For information on the self-registration process, go to the PKI Services Web site at: http://pki.services.ibm.com/expedite/webdocs.shtml.

Once you obtain an X.509 certificate, upload and install it in your z/OS system. The minimum operating system requirement is z/OS 1.4.

gskkyman is a z/OS shell-based utility that is used when importing and managing certificates. For more information, refer to the System Secure Sockets Layer Programming publication.

Registration

If you do not currently have a working connection to Information Exchange, the Help Desk can register your Information Exchange account and user ID with the Certificate Authority and mail you the user number and challenge tokens that you need to retrieve your certificate.

Self-registration

To self-register for an X.509 certificate, you send a message from your Information Exchange account to the self-registration service: USA.IBM1.SELFREG. Make sure that you use Message Class: FFMSG0001. The message text should be plain text, and formatted as shown below:

```
/register SYS.ACCOUNT.USER
contact_name     John Doe
contact_dept     EDI
contact_company  Acme Inc
contact_street   PO Box 1, Acme Drive
contact_city     Tampa
contact_zip      33601
contact_country  US
contact_tel      8135551234
contact_email    john@acme.org
/end
```

NOTE: **`/register`** must be typed in lowercase letters in the message. The remaining items are not case sensitive.

Shortly after you send the above request, you will receive a message in your Information Exchange mailbox. This message contains two pieces of information: your user number and a challenge token.

Retrieval

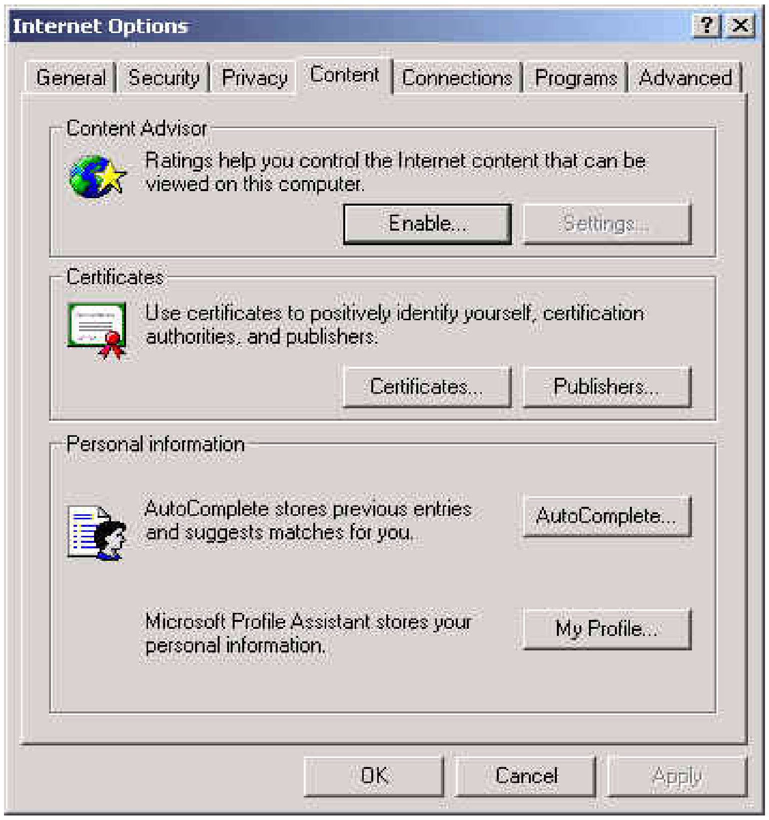
Once you have your user number and challenge token, return to the PKI Services Web site and follow the instructions to retrieve your certificate. The minimum supported browser levels are Netscape 4.7 and Internet Explorer 5. The sample screens in this chapter were captured using Internet Explorer 6.0.

The PKI Services Web site provides step-by-step instructions for generating a private key, and for retrieving and storing the signed certificate.

Once you have successfully performed those steps, proceed to the next section, “Exporting the key from Internet Explorer” which explains how to extract the necessary files. After uploading your certificate, you have two choices for storing it. You can either store your certificate in RACF or a gskkyman-managed HFS file known as a key database. It is recommended that you use RACF. Both procedures are explained in this chapter.

Exporting the key from Internet Explorer

1. In your browser, click **Tools** > **Internet Options**, and then click the **Content** tab. You can also access this dialog by choosing **Internet Option**s on the Windows Control Panel.


Internet Options

General | Security | Privacy | Content | Connections | Programs | Advanced

Content Advisor

Ratings help you control the Internet content that can be viewed on this computer.

Certificates

Use certificates to positively identify yourself, certification authorities, and publishers.

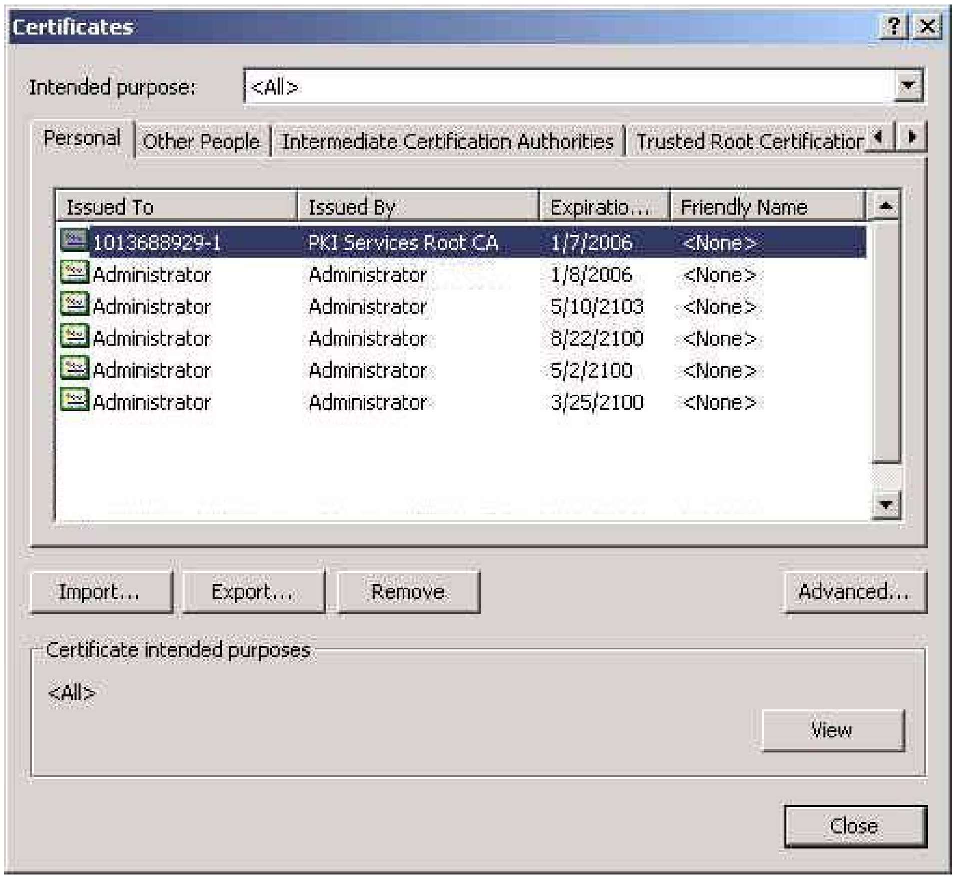
Personal information

AutoComplete stores previous entries and suggests matches for you.

Microsoft Profile Assistant stores your personal information.

2. Click **Certificates**.

A page similar to the one below opens.



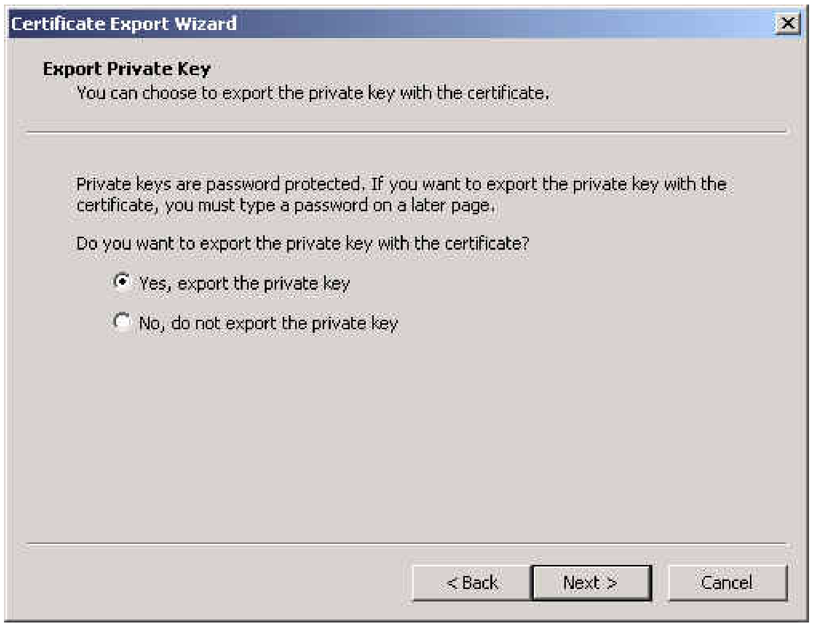
Issued To	Issued By	Expiratio...	Friendly Name
1013688929-1	PKI Services Root CA	1/7/2006	<None>
Administrator	Administrator	1/8/2006	<None>
Administrator	Administrator	5/10/2103	<None>
Administrator	Administrator	8/22/2100	<None>
Administrator	Administrator	5/2/2100	<None>
Administrator	Administrator	3/25/2100	<None>

3. Select the certificate that you just received, and click **Export**.

NOTE: Nothing in the certificate indicates that this certificate is for use with Information Exchange. Your certificate has a different Issued To ID, but it uses a similar naming convention.

The Certificate Export Wizard - Export Private Key page opens.

4. Select **Yes, export the private key** to export the private key along with your certificate.



Export Private Key
You can choose to export the private key with the certificate.

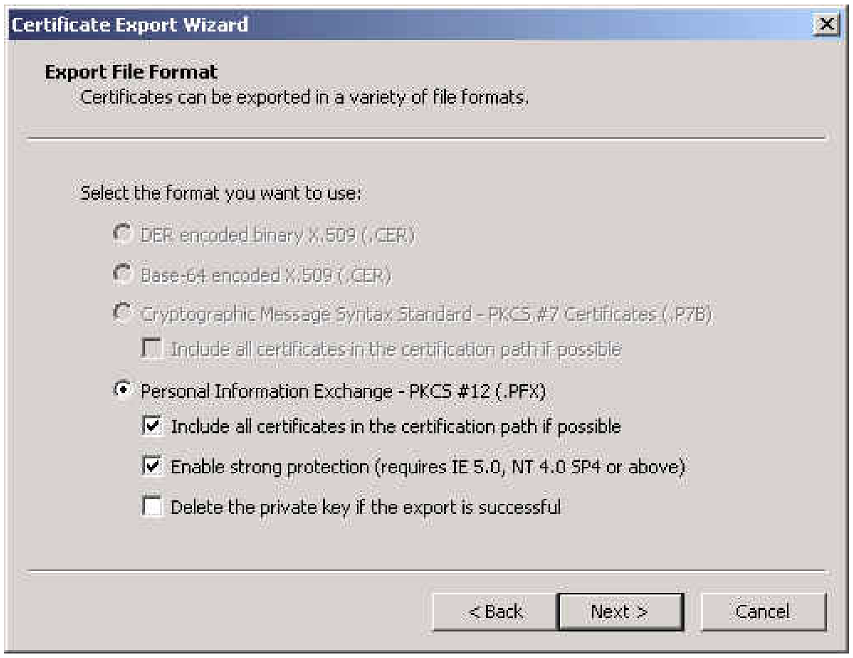
Private keys are password protected. If you want to export the private key with the certificate, you must type a password on a later page.

Do you want to export the private key with the certificate?

● Yes, export the private key
○ No, do not export the private key

5. Click **Next**.

   The Certificate Export Wizard - Export File Format page opens.



Export File Format
Certificates can be exported in a variety of file formats.

Select the format you want to use:

○ DER encoded binary X.509 (.CER)
○ Base-64 encoded X.509 (.CER)
○ Cryptographic Message Syntax Standard - PKCS #7 Certificates (.P7B)
☐ Include all certificates in the certification path if possible
● Personal Information Exchange - PKCS #12 (.PFX)
☑ Include all certificates in the certification path if possible
☑ Enable strong protection (requires IE 5.0, NT 4.0 SP4 or above)
☐ Delete the private key if the export is successful

6. Export the root CA certificate by selecting **Include all certificates in the certification path if possible.**

   NOTE: If you do not export the this certificate as well, you will not be able to import your certificate later.

7. Complete the remaining export wizard steps.
8. Transfer the resulting .pfx file to a z/OS mounted HFS file. Make note of the name and location of the file for later use. It is recommended that you use FTP to transfer the pfx file. Those instructions are not included in this publication.

Using RACF to manage certificates

Using RACF may be more secure than just storing the certificates in a key database. If you use RACF with Expedite Base/MVS, you do not need to provide the KEYRINGPASSWORD or the KEYRINGSTASHFILE parameters.

NOTE: Storing the certificates in a key database is a preliminary step for using RACF management.

Managing certificates using gskkyman

Even if you plan to use RACF to manage your certificates, it is recommended that you use the gskkyman utility to create a key database file, initially store the certificates, and then export the certificates into RACF. For further information, see “Storing certificates in RACF” on page 193.

It is also possible to create a stash file to store your passwords when you use a key database.For more information, refer to the System Secure Sockets Layer Programming publication.

Storing the certificate in a key database

You can store your certificate in a key database. This database resides in an HFS file. You manage the database by logging on directly to UNIX System Services (USS) and using the gskkyman utility. This method of storage is potentially easier for some users than managing certificates in RACF. However, this method is also considered less secure and, therefore, is not recommended.

The first step is to log on to USS. You will use the IBM-supplied program gskkyman to manage your keys and certificates. A sample session is shown below.

Creating a key database

1. From USS, invoke the gskkyman utility by typing **`gskkyman`**.

   The Database Menu displays.

```
     Database Menu

1 - Create new database
2 - Open database
3 - Change database password
4 - Change database record length
5 - Delete database

0 - Exit program

Enter option number: 1
Enter key database name (press ENTER to return to menu): ExpKeyDB.kdb
Enter database password (press ENTER to return to menu):
Re-enter database password:
Enter password expiration in days (press ENTER for no expiration):
Enter database record length (press ENTER to use 2500):
```

2. On the **Enter option number** line, type **1.**
3. Replace the key database name displayed with the name of your key database. This field is case sensitive, so make sure to type the name correctly. For example, you might type **ExpKeyDB.kdb**.
4. Type the database password.
5. Type the database password again to confirm it.
6. Type the password expiration in days or press Enter if the password does not expire.
7. Type the database record length or press Enter to use 2500 characters as the record length.
8. Press Enter.

   The following message displays: `Key database /u/user1/ExpKeyDB.kdb created.`
9. Press Enter to continue.

   The key database is created. Continue with the steps in the next section.

Importing your certificate

Once you have created the key database, you are ready to import your certificate into it. You will need the name and location of the pfx file that you sent by FTP to your z/OS machine.

When you press Enter in Step 9 of the previous procedure, the Key Management Menu displays.

```
         Key Management Menu

         Database: /u/user1/ ExpKeyDB.kdb

    1 - Manage keys and certificates
    2 - Manage certificates
    3 - Manage certificate requests
    4 - Create new certificate request
    5 - Receive certificate issued for your request
    6 - Create a self-signed certificate
    7 - Import a certificate
    8 - Import a certificate and a private key
    9 - Show the default key
   10 - Store database password
   11 - Show database record length

   0 - Exit program

 Enter option number (press ENTER to return to previous menu): 8
 Enter import file name (press ENTER to return to menu):keytest.pfx
 Enter import file password ((press ENTER to return to menu):
 Enter label (press ENTER to return to menu): ExpditeCert
```

1. Type **8** and press Enter.
2. Type the import file name. This is the name you used when you sent the file by FTP to your z/OS system.
3. Type the import file password.
4. Type the certificate label, such as ExpditeCert, and then press Enter.

   The following message displays: `Certificate and key importe`d. Continue with the instructions in the next section.

Setting the default certificate

You must set the certificate that you just imported as the default certificate.

1. On the Key Management Menu, type **1** and press Enter.

```
       Key Management Menu

       Database: /u/user1/ ExpKeyDB.kdb

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive certificate issued for your request
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 1
```

The Key and Certificate List screen displays.

```
       Key and Certificate List

       Database: /u/user1/ ExpKeyDB.kdb

   1 - ExpditeCert

   0 - Return to selection menu

Enter label number (ENTER to return to selection menu, p for previous list): 1
```

2. Type **1** and press Enter.

The Key and Certificate Menu displays.

```
       Key and Certificate Menu

       Label: ExpditeCert

   1 - Show certificate information
   2 - Show key information
   3 - Set key as default
   4 - Set certificate trust status
   5 - Copy certificate and key to another database
   6 - Export certificate to a file
   7 - Export certificate and key to a file
   8 - Delete certificate and key
   9 - Change label
  11 - Create a certificate renewal request

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 3

Press ENTER to continue.
```

3. Type **3** and press Enter.

   The following message displays: `Default key set`.

4. Press Enter.

   The Set Certificate Trust Status screen displays.

```
     SET CERTIFICATE TRUST STATUS

       Key and Certificate Menu

       Label: ExpditeCert

 1 - Show certificate information
 2 - Show key information
 3 - Set key as default
 4 - Set certificate trust status
 5 - Copy certificate and key to another database
 6 - Export certificate to a file
 7 - Export certificate and key to a file
 8 - Delete certificate and key
 9 - Change label
11 - Create a certificate renewal request

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 4

Enter 1 if trusted, 0 if untrusted (press ENTER to return to menu): 1
```

5. On the **Enter option number** line, type **4**.
6. On the **Enter 1 if trusted** line, type **1** and press Enter.

   The following message displays: `Record updated`.
7. Press Enter to continue.

The certificate is now available for use with Expedite/Base MVS as a key database. To use this certificate as the key database, you must specify the key database in the KEYRINGFILE parameter and the key database password in the KEYRINGPWD parameter on either the IDENTIFY or START commands. These commands and passwords are described in Chapter 7, "Expedite Base/ MVS commands." The code sample below shows an IDENTIFY command using the values described above.

```
identify ieaccount(acct1) ieuserid(user1) iepassword(iepasswd)
   keyringfile(/u/user1/ExpKeyDB.kdb) keyringpassword(keypassword);
```

Storing certificates in RACF

It is recommended that you store X.509 keys and certificates in RACF. This section explains how to accomplish that. The instructions assume that you have already obtained a signed certificate and exported it from Internet Explorer (or other workstation X.509 repository) using the steps described earlier in this chapter.

Exporting a user certificate and key using gskkyman

After completing the steps in the previous section, your certificate is stored in a binary format. RACF cannot import a binary certificate; certificates must be Base64 encoded. One method to obtain a Base 64 encoded certificate uses an HFS file as in intermediary step. Not all steps are documented in this example. You may need to select options off other menus to get to the panel shown.

You must export your certificate and private key in a format that is usable by the RACF import facility, as follows:

1. Invoke the gskkyman utility.

The Key Management Menu displays.

```
       Key Management Menu

       Database: /u/user1/ ExpKeyDB.kdb

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive certificate issued for your request
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 1
```

2. Type **1** and press Enter.

   The Key and Certificate List screen displays.

```
      Key and Certificate List

      Database: /u/user1/ ExpKeyDB.kdb

   1 - ExpditeCert

   0 - Return to selection menu
Enter label number (ENTER to return to selection menu, p for previous list): 1
```

3. Type **1** and press Enter.

The Key and Certificate Menu displays.

```
        Key and Certificate Menu

        Label: ExpditeCert

    1 - Show certificate information
    2 - Show key information
    3 - Set key as default
    4 - Set certificate trust status
    5 - Copy certificate and key to another database
    6 - Export certificate to a file
    7 - Export certificate and key to a file
    8 - Delete certificate and key
    9 - Change label
   11 - Create a certificate renewal request

    0 - Exit program

 Enter option number (press ENTER to return to previous menu): 7

 Press ENTER to continue.
```

4. Type **7** and press Enter.

   The Export File Format screen displays.

```
        Export File Format

 1 - Binary PKCS #12 Version 1
 2 - Base64 PKCS #12 Version 1
 3 - Binary PKCS #12 Version 3
 4 - Base64 PKCS #12 Version 3

Select export format (press ENTER to return to menu): 2
Enter export file name (press ENTER to return to menu): expediteCert.b64
Enter export file password (press ENTER to return to menu):
Re-enter export file password:
Enter 1 for strong encryption, 0 for export encryption: 1
Press ENTER to continue.
```

5. On the **Select export format** line, type **2**.
6. On the **Enter export file name** line, type the name of the certificate file that you want to export to Base64 format, such as expediteCert.b64.
7. On the **Enter export file password** line, type the file password.
8. On the **Re-enter export file password** line, type the password again.
9. On the encryption option line, type **1**.
10. Press Enter.

The following message displays: `Certificate and key exported`.

11. Press Enter to continue.

Copying exported files to a data set

There are many ways to copy the exported file from the HFS structure into a sequential (recfm=PS) file that is accessible by RACF. One popular method is using FTP. Another very user-friendly method is to use the ISHELL ISPF facility. In the following example, the OGET command is used.

Type the command as shown below and press Enter. The file is copied.

```
       ISPF Command Shell
Enter TSO or Workstation commands below:
===> oget '/u/user1/expediteCert.b64' expedite.cert.b64
Place cursor on choice and press enter to Retrieve command
=>
=>
=>
```

Importing the certificate and key into RACF

The following instructions assume that you will be using the RACF panels in ISPF to import the certificate and key into RACF. To get to the panel shown below, follow the prompts to perform Digital Certificate Services and *add* your certificate.

1. On the RACF - Digital Certificate Services Main Panel, type **1** on the **Option** line at the top of the screen.

```
        RACF - Digital Certificate Services Main Panel
OPTION ===> 1
                      Personal                Certificate
                      (user ID)      Site     Authority
 For Certificate Type ________  or  _    or  _
 Select one of the following options:
 1. Add a digital certificate to the RACF database.
 2. List information for certificates and be given the
    opportunity to change or delete them.
 3. List a certificate using a filter and be given the opportunity
    to change the trust status, label, or delete it.
 4. Check whether a digital certificate has been added to the
    RACF database and associated with a user ID, by entering a
    data set name:  ____________________________________________
    Password for PKCS12 format data set(in quotes):
    =>                                                        <=
    =>                                                        <=
    =>                                                        <=
    =>                                                        <=
COMMAND ===>
```

The digital certificate description fields display.

```
Enter the name of the data set containing the digital certificate
to be associated with: USER1

Data set name:  ‘expedite.cert.b64’___________________ (in quotes)
Label name:  'ExpediteCert'________________________  (in quotes)

Status: Trust  T    ( N = NOTRUST, T = TRUST, H = HIGHTRUST)

Enter any character to use IBM's crypto service provider:
   _  Integrated Cryptographic Support Facility (ICSF).

Password for PKCS12 format data set(in quotes):
=>                                                              <=
```

2. In the **Data set name** field, type the name of the data set, enclosing the name in quotes.
3. In the **Label name** field, type the label for the data set, enclosing the label in quotes.
4. In the **Status** field, type T (Trust).
5. For the **crypto service provider** field, type any character or leave blank.
6. On the **Password** line, type the password for the data set, enclosing the password in quotes.
7. Press Enter.

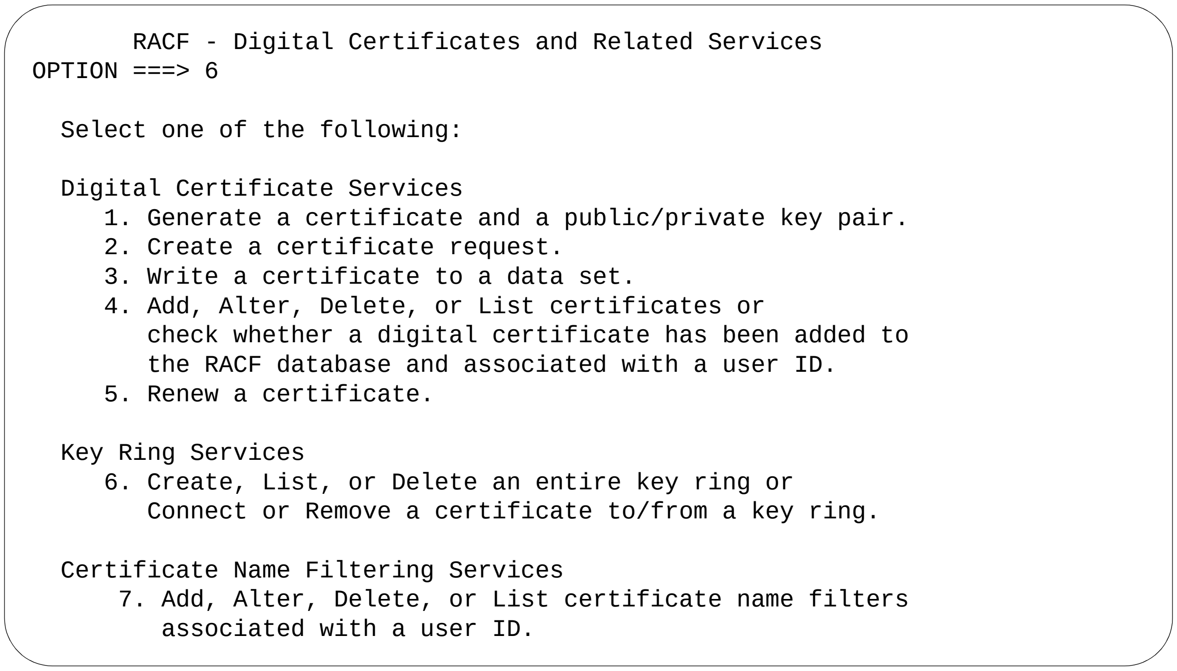
   The certificate is added.

If this is the first time that the Root Certificate Authority (CA) is presented to your system, the following message displays: `Root Certificate Authority not currently defined to RACF. Top CERTAUTH certificate added with the trust status`. This is important to know later as you will be required to connect the root certificate to the key ring. If the root certificate already resides on your system, you will need to find the label associated with it in order to connect it to your key ring. Continue with the instructions in the next section.

Creating a user key ring

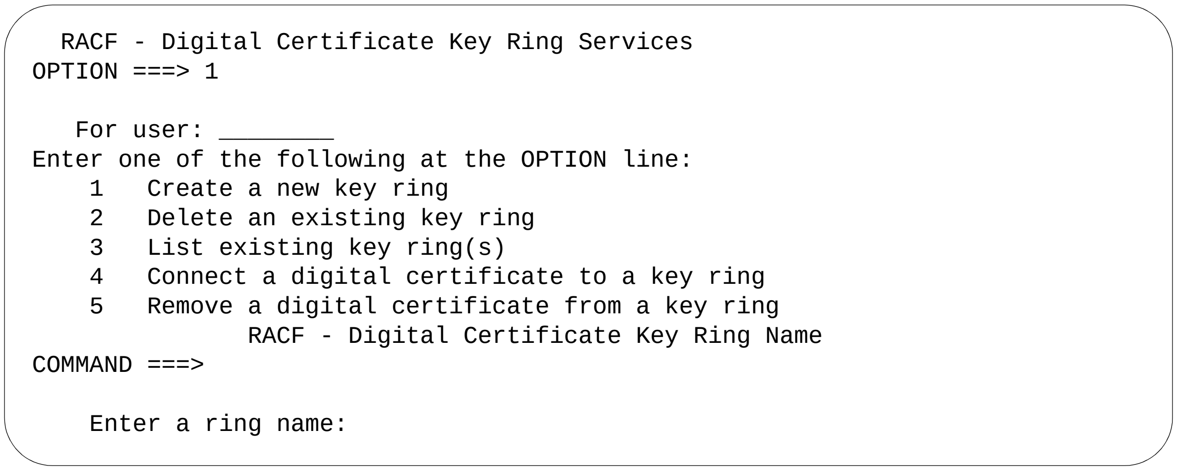
Next, you must create a key ring for your user ID.

1. On the RACF - Digital Certificates and Related Services screen, type **6** on the **Option** line at the top of the screen.



```
       RACF - Digital Certificates and Related Services
OPTION ===> 6

  Select one of the following:

  Digital Certificate Services
     1. Generate a certificate and a public/private key pair.
     2. Create a certificate request.
     3. Write a certificate to a data set.
     4. Add, Alter, Delete, or List certificates or
        check whether a digital certificate has been added to
        the RACF database and associated with a user ID.
     5. Renew a certificate.

  Key Ring Services
     6. Create, List, or Delete an entire key ring or
        Connect or Remove a certificate to/from a key ring.

  Certificate Name Filtering Services
      7. Add, Alter, Delete, or List certificate name filters
         associated with a user ID.
```

The user detail fields display.



```
  RACF - Digital Certificate Key Ring Services
OPTION ===> 1

   For user: ________
Enter one of the following at the OPTION line:
    1   Create a new key ring
    2   Delete an existing key ring
    3   List existing key ring(s)
    4   Connect a digital certificate to a key ring
    5   Remove a digital certificate from a key ring
               RACF - Digital Certificate Key Ring Name
COMMAND ===>

    Enter a ring name:
```

2. On the **Option** line at the top of the screen, type **1**.

```
Enter a ring name:
 ExpediteRing__________________________________________________
 ____________________________________________________________
 ____________________________________________________________
 _________________________________________________________
A ring name may not be a single asterisk *  and
blanks are not allowed.
```

3. On the **Enter a ring name** line, type the name of the ring, following the rules displayed below the field.

4. Press Enter.

   The key ring is created.

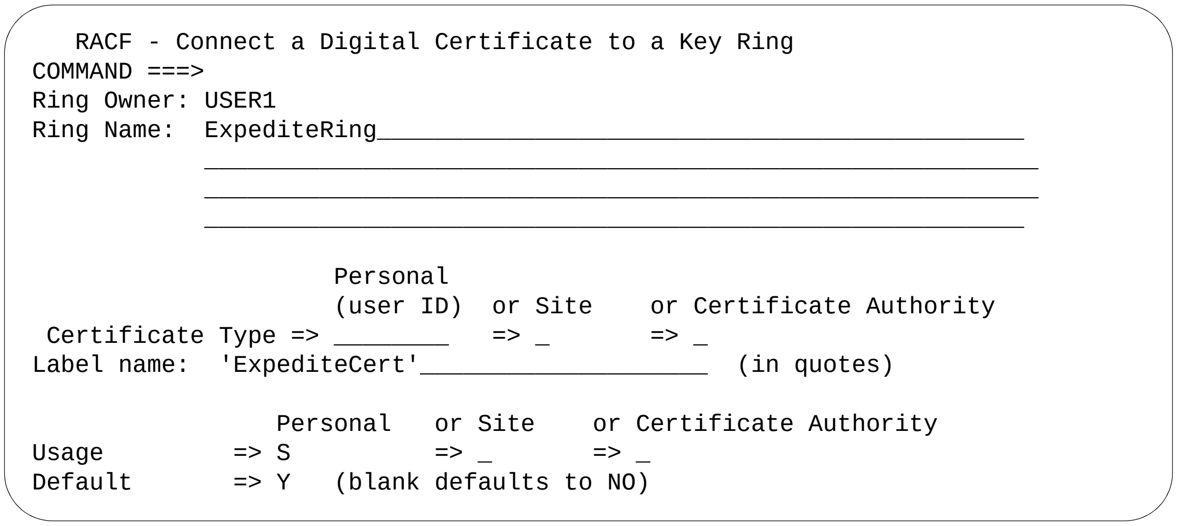
Connecting the certificates to the user

Finally, you must connect your imported certificates to the user's key ring. If you followed the instructions provided earlier in this section, when you imported the certificate and private key, the Certificate Authority certificate was also imported.

1. On the RACF - Digital Certificate Key Ring Services screen, type **4** on the **Option** line at the top of the screen.

```
           RACF - Digital Certificate Key Ring Services
OPTION ===> 4
Key Ring ExpediteRing has been successfully added
   For user: ________
Enter one of the following at the OPTION line:
   1   Create a new key ring
   2   Delete an existing key ring
   3   List existing key ring(s)
   4   Connect a digital certificate to a key ring
   5   Remove a digital certificate from a key ring
```

The key ring detail screen displays.



```
   RACF - Connect a Digital Certificate to a Key Ring
COMMAND ===>
Ring Owner: USER1
Ring Name:  ExpediteRing_____________________________________________
            __________________________________________________________
            __________________________________________________________
            _________________________________________________________


                     Personal
                     (user ID)  or Site    or Certificate Authority
 Certificate Type => ________   => _       => _
Label name:  'ExpediteCert'____________________  (in quotes)

                 Personal   or Site    or Certificate Authority
Usage          => S         => _       => _
Default        => Y   (blank defaults to NO)
```

2. In the **Ring Name** field, type the name of the ring.
3. In the **Label Name** field, type the label of the certificate to connect, enclosing it in quotes.
4. In the **Usage** field, type **S** under **Personal**.
5. In the **Default** field, type **Y**.
6. Press Enter.

   The following message displays: `Certificate successfully connected to key ring`.

Listing root certificate authorities

The following steps describe how to identify the label that was used for the Expedite Certificate authority. To locate the correct certificate, you must list the certificates.

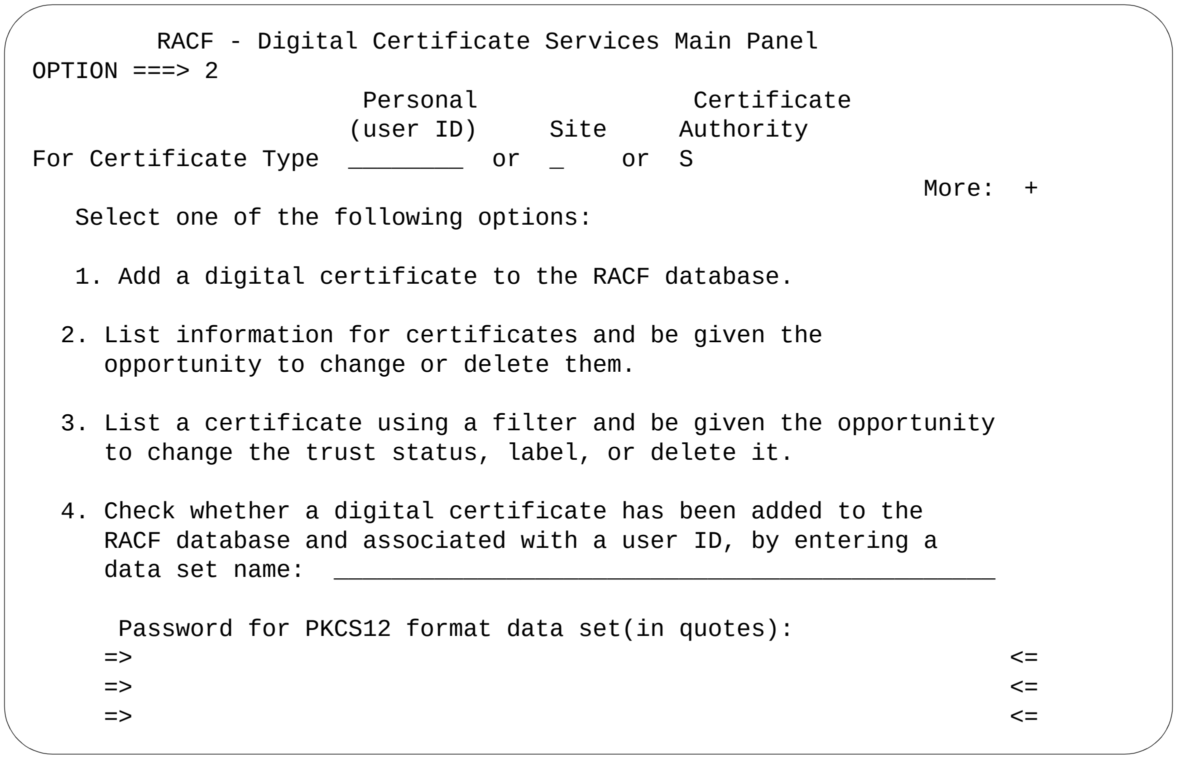
1. On the RACF - Digital Certificates and Related Services screen, type **4** on the **Option** line at the top of the screen.

```
           RACF - Digital Certificates and Related Services
OPTION ===> 4

  Select one of the following:

     Digital Certificate Services
        1. Generate a certificate and a public/private key pair.
        2. Create a certificate request.
        3. Write a certificate to a data set.
        4. Add, Alter, Delete, or List certificates or
           check whether a digital certificate has been added to
           the RACF database and associated with a user ID.
        5. Renew a certificate.

     Key Ring Services
        6. Create, List, or Delete an entire key ring or
           Connect or Remove a certificate to/from a key ring.

     Certificate Name Filtering Services
        7. Add, Alter, Delete, or List certificate name filters
                         associated with a user ID.
```

   The RACF - Digital Certificates Services Main Panel displays

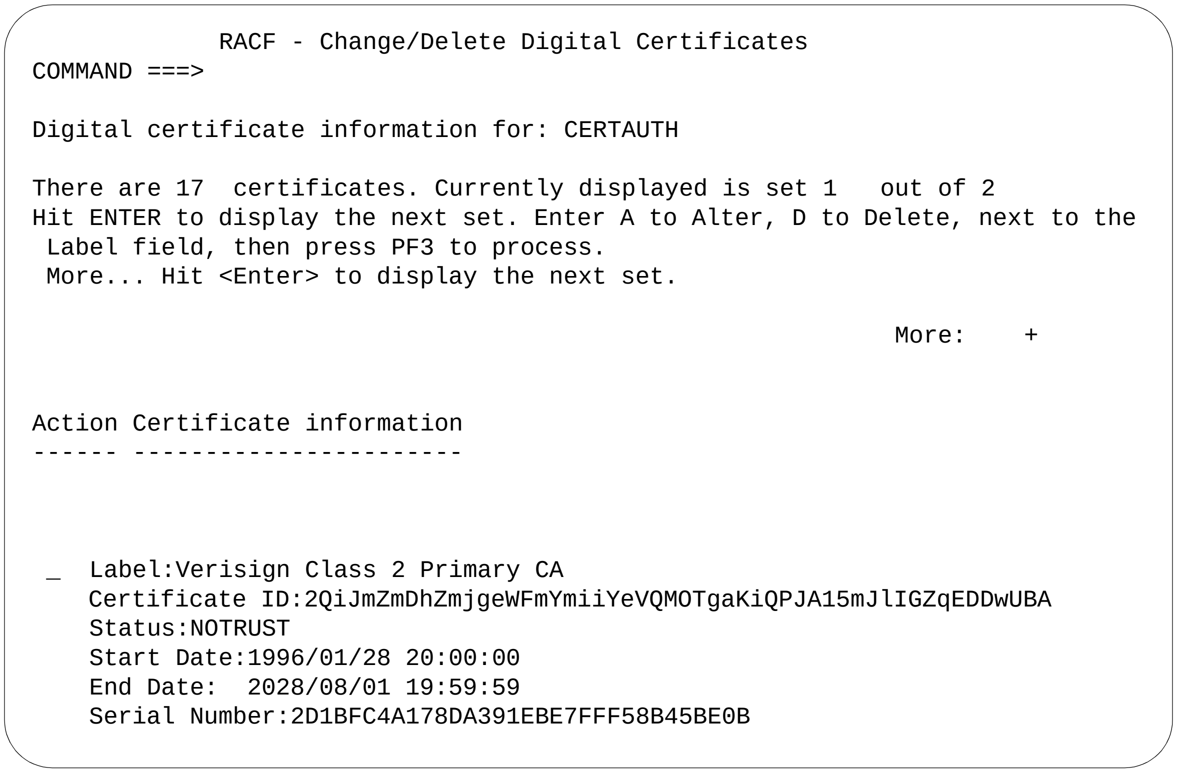
2. On the **Option** line, type **2**.

3. On the **For Certificate Type** line, type **S** under **Certificate Authority**.



```
          RACF - Digital Certificate Services Main Panel
OPTION ===> 2
                       Personal               Certificate
                      (user ID)     Site     Authority
For Certificate Type  ________  or  _    or  S
                                                          More:  +
   Select one of the following options:

   1. Add a digital certificate to the RACF database.

  2. List information for certificates and be given the
     opportunity to change or delete them.

  3. List a certificate using a filter and be given the opportunity
     to change the trust status, label, or delete it.

  4. Check whether a digital certificate has been added to the
     RACF database and associated with a user ID, by entering a
     data set name:  ______________________________________________

      Password for PKCS12 format data set(in quotes):
     =>                                                              <=
     =>                                                              <=
     =>                                                              <=
```

The RACF - Change/Delete Digital Certificates screen displays.



```
             RACF - Change/Delete Digital Certificates
COMMAND ===>

Digital certificate information for: CERTAUTH

There are 17  certificates. Currently displayed is set 1   out of 2
Hit ENTER to display the next set. Enter A to Alter, D to Delete, next to the
 Label field, then press PF3 to process.
 More... Hit <Enter> to display the next set.

                                                            More:    +


Action Certificate information
------ -----------------------



 _   Label:Verisign Class 2 Primary CA
     Certificate ID:2QiJmZmDhZmjgeWFmYmiiYeVQMOTgaKiQPJA15mJlIGZqEDDwUBA
     Status:NOTRUST
     Start Date:1996/01/28 20:00:00
     End Date:  2028/08/01 19:59:59
     Serial Number:2D1BFC4A178DA391EBE7FFF58B45BE0B
```

4. Press Enter to display Set 2.

```
             RACF - Change/Delete Digital Certificates
COMMAND ===>

Digital certificate information for: CERTAUTH

There are 17  certificates. Currently displayed is set 2   out of 2
Hit ENTER to display the next set. Enter A to Alter, D to Delete, next to the
 Label field, then press PF3 to process.
 ***LAST SET CURRENTLY DISPLAYED***
 You may hit <Enter> to go back to the first set.
                                                                More:     +


Action Certificate information
------ -----------------------



 _   Label:Verisign International Svr CA
     Certificate ID:2QiJmZmDhZmjgeWFmYmiiYeVQMmVo4WZlYGjiZaVgZNA4qWZQMPB
     Status:NOTRUST
     Start Date:1997/04/16 20:00:00
     End Date:  2011/10/24 19:59:59
     Serial Number:254B8A853842CCE358F8C5DDAE226EA4
```

5. Scroll down by pressing F8 until you see the following:

   ```
   Issuer's Name:CN=PKI Services Root CA.OU=Interchange Services for
   e-business.O=IBM.C=US
   ```

   PKI Services Root is the Certificate Authority for Expedite certificates.

6. Press F7 once to find the label for this certificate. That label is used in the following step.

The RACF - Connect a Digital Certificate to a Key Ring screen displays.

```
       RACF - Connect a Digital Certificate to a Key Ring
COMMAND ===>
Ring Owner: USER1
Ring Name:  ExpediteRing________________________________________

            __________________________________________________

            __________________________________________________

            __________________________________________________

                     Personal
                     (user ID  or Site    or Certificate Authority
 Certificate Type => ________  => _       => _
Label name:  ___________________________________  (in quotes)

                 Personal   or Site    or Certificate Authority
Usage         =>            => _       => S
Default       => _   (blank defaults to NO)
```

7. In the **Label Name** field, type the label name for the Root Certificate Authority for the key ring enclosing in quotes.

8. In the **Certificate type** field, type **S** under Certificate Authority.

9. In the **Usage** field, type **S** under Certificate Authority.

10. Press Enter.

    A message indicating that the operation is successful is displayed.

11. Exit RACF.

    This process is complete.

The certificate is now available for use with Expedite/Base MVS as a key ring. To use this certificate as the key ring, you must specify the key ring in the KEYRINGFILE parameter in the KEYRINGPWD parameter on either the IDENTIFY or START commands. These commands are described in Chapter 7, "Expedite Base/MVS commands." The code sample below shows an IDENTIFY command using the values described above.

```
identify ieaccount(acct1) ieuserid(user1)iepassword(iepasswd)
keyringfile(user1/ExpediteRing);
```

Sample JCL for use with SSL

A sample of the JCL used to start Expedite Base/MVS using SSL is provided here. This example uses Session-level recovery. All parameters that you must enter exactly are provided in the following example using uppercase letters. JCL variables specific to your system or a job are shown in lowercase letters.

```
//jobname JOB (insert your installation-specific job statement)
//JOBLIB DD DSN=expedite.mvs.loadlib,DISP=SHR
//expstep EXEC PGM=IEBASE,REGION=0K,PARM='POSIX(ON)/'
//SYSPRINT DD sysout=*
//SYSOUT DD sysout=*
//*
//*command files
//INMSG DD DSN=user1.message.command,DISP=SHR
//INPRO DD DSN=user1.profile.command,DISP=SHR
//*
//*response files
//OUTMSG DD DSN=user1.message.response,DISP=OLD
//OUTPRO DD DSN=user1.profile.response,DISP=OLD
//*
//*trace files
//BASETRC DD DSN=user1.iebase.trace,DISP=OLD
//LINKTRC DD DSN=user1.line.trace,DISP=OLD
//*
//*error text files
//ERRORMSG DD DSN=expedite.error.messages(EXXMSG),DISP=SHR
//ERRORTXT DD DSN=expedite.error.messages(EXXTXT),DISP=SHR
```

Sample INPRO commands necessary to use SSL

The following sample shows the INPRO commands necessary to communicate SSL and assumes use of a RACF key ring. The TCPCOMM command reflects the internet IP address for use with SSL in the US. Customers can determine the correct IP address used in their geography by contacting their local Information Exchange help desk.

```
identify ieaccount(acct1) ieuserid(user1) iepassword(iepasswd)
keyringfile(user1/ExpediteRing);
ssl enablessl(y);
transmit commtype(t);
tcpcomm ietcphost1(204.146.24.79) ietcpport1(3667);
```

Product examples

The following sections contain three Expedite Base/MVS jobs. The jobs send and receive files, EDI data, and stream data. They also retrieve audit data, receive Information Exchange system error messages, and cancel previously sent files. The examples shown in this appendix are also included on the product tape.

Example 1

The following example defines a list, sends a file, and receives a file. It also retrieves audit data and places it in the Information Exchange mailbox, cancels previously sent files, and receives Information Exchange system error messages. It uses session-level recovery.

```
//jobname JOB (insert your installation-specific job statement)
//JOBLIB DD DSN=exxhlq.AGXQ461.SEXXLOAD,DISP=SHR
//EXP EXEC PGM=IEBASE,REGION=0K
//INMSG DD *
#Define a temporary list named mylist containing two users #

list listname(mylist)
account(act1) userid(user002) account(act1) userid(user003);

#Send a file named user01.test.data to the list named mylist #
#use user class testclas. #

send fileid(user01.test.data) listname(mylist) class(testclas);

#Receive a file called user01.test.out from account act1, #
#userid user002, and user class testclas #

receive fileid(user01.test.out) account(act1)
 userid(user002) class(testclas);

#Cancel any files sent to account act3, userid user005, #
#with a class of testcls #

cancel account(act3) userid(user005) class(testcls);

#Load the audit records for any files sent to or received #
#from account act2, userid user005
#The audit data must be retrieved in a subsequent session #
```

```
audit account(act2) userid(user005);

#Receive IE system error messages into the file referenced by #
#DD Name IESYSERR #

receive fileid(dd:iesyserr) account(*system*) userid(*errmsg*);
/*
//INPRO DD *
#Provide the IE account, userid, and password to be used for #
#the IE session start #

identify ieaccount(act1) ieuserid(user01) iepassword(paswrd)
   timezone(est);

#Provide the IE LU Name and the LU name of Expedite Base/MVS #

snacomm ieluname(ieapplid) userluname(myapplid);
/*
//OUTMSG DD DSN=USER01.OUTMSG,DISP=OLD
//OUTPRO DD DSN=USER01.OUTPRO,DISP=OLD
//ERRORMSG DD DSN=exxhlq.AGXQ461.SEXXMSGS(EXXMSG) DISP=SHR
//ERRORTXT DD DSN=exxhlq.AGXQ461.SEXXMSGS(EXXTXT) DISP=SHR
//SYSPRINT DD SYSOUT=*
```

Example 2

The following example sends an EDI file using both an EDI qualifier and EDI destination table. It also receives an EDI file and Information Exchange system error messages. It uses checkpoint-level recovery.

```
//jobname JOB (insert your installation-specific job statement)
//JOBLIB  DD DSN=exxhlq.AGXQ461.SEXXLOAD,DISP=SHR
//EXP     EXEC PGM=IEBASE,REGION=0K
//INMSG   DD *
#Send the EDI data in file user01.edi.test. #
#Use a user class of editest #

sendedi fileid(user01.edi.test) class(editest);

#Receive the EDI data sent from account act1, userid user002 #
#with a user class of editest #
#Place the data in file user01.edi.out #

receiveedi fileid(user01.edi.out)
account(act1) userid(user002) class(editest);

#Receive IE system error messages into the file referenced by #
#DD Name IESYSERR #

receive fileid(dd:iesyserr) account(*system*) userid(*errmsg*);
//INPRO DD *
#Provide the IE account, user ID, and password to be used for #
#the IE session start #

identify ieaccount(act1) ieuserid(user01) iepassword(paswrd)
   timezone(est);

#Provide the IE LU name and the LU name of Expedite Base/MVS #

snacomm ieluname(ieapplid) userluname(myapplid);

#Ask for checkpoint-level recovery #
```

```
transmit recovery(c);
//QUALTBL DD *
DATATYPE(X) QUALIFIER(01) ALIAS(GX01) TTABLE(DD:TTABLE01);
/*
//TTABLE01 DD *
EDIDEST(DUNS01) ACCOUNT(ACT2) USERID(USER001);
EDIDEST(DUNS.2) ACCOUNT(ACT3) USERID(USER005);
/*
//OUTMSG   DD DSN=USER01.OUTMSG,DISP=OLD
//OUTPRO   DD DSN=USER01.OUTPRO,DISP=OLD
//ERRORMSG DD DSN=exxhlq.AGXQ461.SEXXMSGS(EXXMSG) DISP=SHR
//ERRORTXT DD DSN=exxhlq.AGXQ461.SEXXMSGS(EXXTXT) DISP=SHR
//IESYSERR DD DSN=USER01.IEERRORS,DISP=OLD
//OUTWORK  DD DSN=USER01.OUTWORK,DISP=OLD
//SESSION  DD DSN=USER01.SESSION,DISP=OLD
//RCVWORK  DD DSN=USER01.RCVWORK,DISP=OLD
//SYSPRINT DD SYSOUT=*
```

Example 3

The following example sends data from the message command file (INMSG) using the SENDSTREAM command. It also receives data and Information Exchange system error messages into the message response file (OUTMSG) using RECEIVESTREAM commands. It uses session-level recovery.

```
//jobname JOB (insert your installation-specific job statement)
//JOBLIB  DD DSN=exxhlq.AGXQ461.SEXXLOAD,DISP=SHR
//EXP     EXEC PGM=IEBASE,REGION=0K
//INMSG   DD *
#Send data following the SENDSTREAM command to #
#jones in alias table ptb1 with user class testclas #
#the data ends when the xxx string is found #

sendstream alias(ptb1) aliasname(jones) class(testclas) endstr(xxx);
line one of data
line two of data
xxx

#Receive data from account act1, userid user002 with #
#user class class1 #
#End the data with the string xxxx #
#Data received with the receivestream command follows #
#the RECEIVED record in OUTMSG #

receivestream account(act1) userid(user002) endstr(xxxx) class(class1);

#Receive IE system error messages #

receivestream account(*system*) userid(*errmsg*) endstr(?x$xx?);
/*
//INPRO   DD *
#Provide the IE account, userid, and password to be used for #
#the IE session start #

identify ieaccount(act1) ieuserid(user01) iepassword(paswrd)
   timezone(est);

#Provide the IE LU Name and the LU name of Expedite Base/MVS #

snacomm ieluname(ieapplid) userluname(myapplid);
```

```
//OUTMSG DD SYSOUT=*
//OUTPRO DD SYSOUT=*
//ERRORMSG DD DSN=exxhlq.AGXQ461.SEXXMSGS(EXXMSG) DISP=SHR
//ERRORTXT DD DSN=exxhlq.AGXQ461.SEXXMSGS(EXXTXT) DISP=SHR
//SYSPRINT DD SYSOUT=*
```

Common data header

Information Exchange interfaces can use a common data header (CDH) to communicate detailed information about files and messages to other interfaces and to Information Exchange. Expedite Base/MVS builds a CDH for every file sent, so you never need to build a CDH yourself. Expedite Base/MVS also recognizes CDHs received from other interfaces.

The CDH provides details (such as file name and carriage-return and line-feed options) that let the receiving interface reconstruct a received message into its original format. It also makes more information available to the recipient of a file.

The actual location and format of the CDH is transparent to the Expedite Base/MVS user. However, when looking at a trace of the actual Information Exchange message flow, you might notice an additional message at the beginning of each Information Exchange message group. This extra message contains the CDH.

The information in the CDH is presented to you in the form of parameters in the RECEIVED or AVAILABLE records.

Construction of the CDH

The CDH is constructed during the SEND, SENDEDI, SENDSTREAM, and PUTMEMBER process and contains the following information:

- File name and location
- Record format and length
- Data type (EBCDIC or binary)
- Record delimiter (carriage-return line-feed characters or length indicators)
- Data format (EDIFACT, X12, UCS, or UN/TDI)
- Description (see “User data” on page 210)
- Unique ID
- Sending code page
- Sending system
- Sending version
- File date
- File time

- Carriage-return line-feed (CRLF) and end-of-file (EOF) characters
- EDI sender ID qualifier
- EDI sender ID
- EDI receiver ID qualifier
- EDI receiver ID
- EDI Interchange Control Number
- Length of data
- Library owning account
- Library name
- Library member
- Library member replacement
- Compression software package name
- Compression software package version
- Name of compressed file

User data

To further identify a file or to include extra information or instructions, you can supply additional user data in the CDH. This feature is optional, but you can add up to 79 bytes of user data by using the DESCRIPTION parameter of the SEND, SENDSTREAM, SENDEDI, and PUTMEMBER commands.

NOTE: For the SENDEDI commands, a CDH is constructed for each interchange contained in the file. If a DESCRIPTION parameter is present, the user data supplied is copied to the CDH for each interchange.

Libraries

The Expedite Base/MVS interfaces use the CDH library fields to pass the library information to Information Exchange when processing a PUTMEMBER or GETMEMBER command. This information includes the library owner's account, the library name, the member name, and a replace indicator. For other commands, these fields will not be included in the CDH.

CDH entry fields

For the layout of the CDH entry fields, see I*nformation Exchange Interface Programming Guide.*

Sample audit report

With the AUDIT command, users can select specific criteria when they process the command (for example, a user might want to have a report only on messages received or only on messages sent). Because of this, the content of any given report depends on the criteria selected. The sample report shown at the end of this appendix illustrates the output that might result if all the criteria are selected. The program used to generate this report is included on the product tape.

NOTE: The procedures and the example presented in this appendix are only for level 1 message audit record formats. Modifications would be necessary to format a level 2 or 3 audit record.

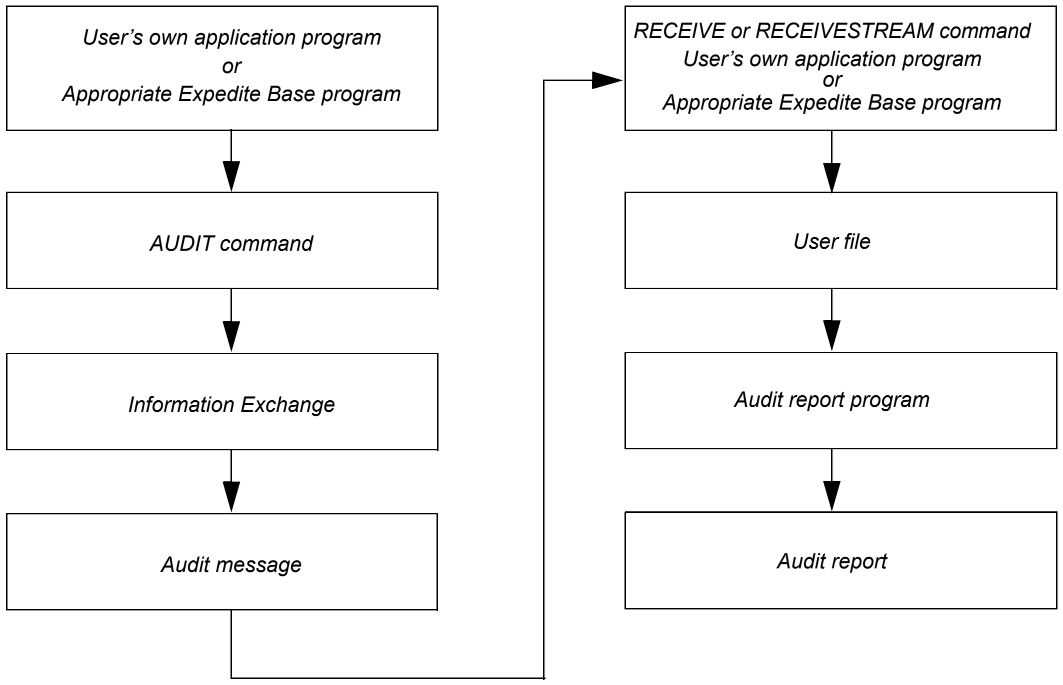
Using the AUDIT command

To use the AUDIT command, follow these steps:

1. Retrieve the audit records by issuing the AUDIT command. This places the records in your mailbox.
2. When the audit information is available, issue the RECEIVE command to retrieve the records from your mailbox. The audit information comes from the *SYSTEM* account ID and the *AUDITS* user ID in message user class #SAUDIT.

   NOTE: Audits are not available immediately; you cannot successfully issue the AUDIT command immediately followed by the RECEIVE command. Audits are usually available during the next session.

3. Run the sample audit report program, EXXAUDIT.


User's own application program
or
Appropriate Expedite Base program
AUDIT command
Information Exchange
Audit message
RECEIVE or RECEIVESTREAM command
User's own application program
or
Appropriate Expedite Base program
User file
Audit report program
Audit report

Sample audit report layout

The following pages illustrate the layout of an audit report showing the status of messages sent and received by one user. The final two pages of the report show totals for each category.

```
DATE:  03 31 04                                                                                   PAGE  000004

                                                INFORMATION EXCHANGE
                                                 SAMPLE AUDIT REPORT

                                      ACCOUNT:  AAAA                    USERID:  USERAA

OUTBOUND TO:               SYSID:         ACCT: ZZZZ     USERID: USERZZ      ALIAS:            EDI QUAL:            EDI RECEIVER:

DATE        TIME        DATE          TIME          MESSAGE     MESSAGE     MESSAGE      TEXT         STATUS       PURCE             EDI
SENT        SENT        RCVD/PURGE    RCVD/PURGE    NAME        SEQUENCE    CLASS        SIZE                      REASON            CONTROL ID

03/19/04    14:02:00    03/19/04      14:03:52      MESSAGE1    12345       CLASS001     00000004     RECEIVED

03/19/04    14:02:25    03/19/04      14:03:55      MESSAGE2    12350       CLASS002     00000009     RECEIVED

03/19/04    14:04:42    03/24/04      17:40:06      MESSAGE3    12487       CLASS003     00000012     PURGED       CANCEL BY RCVR

03/19/04    14:02:52    03/25/04      17:40:27      MESSAGE4    12489       CLASS004     00000025     PURGED       CANCEL BY RCVR

03/19/04    10:12:19    03/20/04      12:10:11      X12MSG      X12         #E2          00000500     RECEIVED                       0000123456789
```

```
DATE:  03 31 04                                                                                   PAGE  000005

                                                INFORMATION EXCHANGE
                                                 SAMPLE AUDIT REPORT

                                      ACCOUNT:  AAAA                    USERID:  USERAA

OUTBOUND TOTALS:                      MSG TO SELF TOTALS;                              INBOUND TOTALS:

IN TRANSIT         0000000000         IN TRANSIT          0000000000                   IN TRANSIT          0000000000

RECEIVED           0000000001         RECEIVED            0000000000                   RECEIVED            0000000003

PURGED             0000000005         PURGED              0000000001                   PURGED              0000000002

IN MAILBOX         0000000001         IN MAILBOX          0000000002                   IN MAILBOX          0000000001
```

```
DATE:  03 31 04                                                                                   PAGE  000006

                                                INFORMATION EXCHANGE
                                                 SAMPLE AUDIT REPORT

                                             TOTALS FOR ACCOUNT:  AAAA

OUTBOUND TOTALS:                      MSG TO SELF TOTALS;                              INBOUND TOTALS:

IN TRANSIT         0000000000         IN TRANSIT          0000000000                   IN TRANSIT          0000000000

RECEIVED           0000000001         RECEIVED            0000000000                   RECEIVED            0000000003

PURGED             0000000005         PURGED              0000000001                   PURGED              0000000002

IN MAILBOX         0000000001         IN MAILBOX          0000000002                   IN MAILBOX          0000000001
```

Expedite Base/MVS messages and codes

This appendix describes the Expedite Base/MVS messages and codes. They are divided into the following categories:

- Expedite Base/MVS completion codes (00 - 12), page 216
- Expedite Base/MVS reason codes (1004 - 1062), page 217
- Expedite Base/MVS return codes (00000 - 32000), page 220

This section includes the following codes:

- Successful completion (00000), page 220
- Message command file syntax errors (2014 - 4988), page 220
- Profile command file syntax errors (5024 - 6202), page 251
- Parser errors (14000 - 15042), page 255
- Destination verification errors (16020 - 16060), page 257
- EDI errors (17102 - 18300), page 258
- TCP/IP communication errors (19006 - 19022), page 263
- File validation errors (20022 - 20064), page 265
- General environment errors (20365 - 23510), page 267
- Session start and end errors (24000 - 24620), page 271
- Session errors (28000 - 28200), page 274
- APPC communications support errors (29800 - 29866), page 276
- SSL communication errors, (30006 - 30119) page 283
- Unexpected errors (31000 - 32000), page 286
- Expedite Base/MVS error message identifiers to the operator console (EXX4003 - EXX4058), page 287

Expedite Base/MVS completion codes

These completion codes appear in your job log as the return code for a step.

00

Explanation: Expedite Base/MVS completed successfully with no errors.

User Response: No user action is required.

04

Explanation: Expedite Base/MVS completed the session, but errors or warnings were detected.

User Response: Check the RETURN and WARNING records in the message response file, OUTMSG, to determine what problems occurred. Try the commands again that caused errors, if necessary.

08

Explanation: Expedite Base/MVS could not complete the session due to an error.

User Response: Check the SESSIONEND record to determine the cause of the error. Correct the JCL and try the program again.

12

Explanation: An invalid parameter was specified on the EXEC statement.

User Response: Check SYSPRINT for an error message. For more information, see “EXEC statement” on page 21. Correct the problem and try the program again.

Expedite Base/MVS reason codes

Reason codes are generally provided when there are problems accessing a file. The reason code is accompanied by one of the error codes described in the subsequent sections of this appendix.

1004 Unable to allocate data set x, return=yy, reason=zzzz.

Explanation: The data set x could not be allocated by Expedite Base/MVS. The return code yy and reason code zzzz are returned by Expedite Base/MVS and give additional information about the cause.

User Response: Check to be sure that the data set is correctly defined in your JCL or command file and that your job can access the data set.

1008 Unable to read file control block for data set x.

Explanation: Expedite Base/MVS was unable to find the file control block for data set x. This is usually caused by a missing DD statement for the data set.

User Response: Check to be sure that the data set is correctly defined in your JCL.

1012 Missing member name in data set x.

Explanation: The data set x is a PDS, but no member name was specified.

User Response: Check to be sure that the data set is correctly defined in your JCL or command file.

1014 Data set x is not a PDS.

Explanation: A member name was specified, but data set x is not a PDS.

User Response: Check to be sure that the data set is correctly defined in your JCL or command file.

1016 Unable to find TIOT control block to access data set x.

Explanation: A system error occurred while trying to access data set x.

User Response: Check to be sure that your JCL is correctly defined and that there are no problems on your MVS system.

1020 Data set x is cataloged but not found.

Explanation: The data set x is cataloged but cannot be found.

User Response: Check to be sure that the data set exists and can be accessed.

1024 Unsupported data set organization for data set x.

Explanation: The data set organization of data set x is not supported by Expedite Base/MVS. Expedite Base/MVS supports only sequential data sets and PDS members.

User Response: Specify either a sequential data set or PDS member in your JCL or command file.

1028 Unsupported device type for data set x.

Explanation: Data set x is on a device that is not supported by Expedite Base/MVS.

User Response: Specify a data set that is on a supported device type.

1032 The open failed for data set *x*.

Explanation: The MVS open macro failed for data set x.

User Response: Check your job log for any MVS system messages. Also check to make sure the data set is correctly defined in your JCL or command file and that your job can access the data set.

1036 The open abended for dataset x, abend code=yyyy, reason=zzzz.

Explanation: The open call for data set x abended. The abend code yyyy and reason code zzzz give additional information on the cause.

User Response: Check to be sure that the data set is correctly defined in your JCL or command file and that your job can access the data set. You can also look up the return code and reason code in the appropriate MVS system codes manual.

1040 Insufficient RACF authority to access data set x.

Explanation: The data set x could not be allocated by Expedite Base/MVS due to insufficient RACF authority.

User Response: Check to be sure that the data set is correctly defined in your JCL or command file, and that your job has the correct authority to access the data set.

1044 Data set x not found.

Explanation: The data set x could not be found.

User Response: Check to be sure that the data set is correctly defined in your JCL or command file. Be sure that there are no spelling errors and that the fully qualified data set name is specified.

1046 Invalid PDS member x specified.

Explanation: The specified PDS exists but does not contain this member.

User Response: Specify an existing PDS member. Be sure that the member name is spelled correctly.

1048 Data set x in use.

Explanation: The data set x could not be accessed because it is currently in use.

User Response: Free the data set, or wait until the data set is no longer being used.

1050 The syntax for fileid x is invalid.

Explanation: The syntax of data set x specified by the FILEID parameter is invalid.

User Response: Correct the FILEID parameter value in your command file.

1052 PDS member has DISP=MOD in file x.

Explanation: DISP=MOD is not permitted for a member of a PDS.

User Response: Correct your JCL or command file and try the program again. For more information on file restrictions, see “Expedite Base/MVS files” on page 12.

1054 PDFS member has spanned records in file x.

Explanation: Spanned records in a PDS member are not supported by Expedite Base/MVS.

User Response: Correct your JCL, your command file, or the file specified, and try the program again. For more information on file restrictions, see “Expedite Base/MVS files” on page 12.

1056 Data set x has RECFM=U, which is not supported.

Explanation: Unformatted records are not supported by Expedite Base/MVS.

User Response: Correct your JCL, your command file, or the file specified, and try the program again. For more information on file restrictions, see “Expedite Base/MVS files” on page 12

1060 Unable to access file x, RC1=AAAA, RC2=YYYY.

Explanation: File x could not be opened. The values for RC1 and RC2 give additional information.

User Response: Check to be sure that the data set is correctly defined in your JCL or command file and that your job can access the data set. If the problem continues, report the problem to Help Desk.

1062 Invalid record length for file x.

Explanation: The record length for the file is invalid. The LRECL must be at least 1 byte for fixed length records and at least 4 bytes for variable length records.

Expedite Base/MVS return codes

This section provides an explanation of the Expedite Base/MVS return codes and tells you what action to take in response to each return code. The product displays these codes in the WARNING, RETURN, PROFILERC, and SESSIONEND records in the response files.

Successful completion

This section describes the return code for successful completion of the process.

00000 Session completed successfully.

Explanation: Expedite completed successfully.

User Response: No user action is required.

Message command file syntax errors

The following section describes the return codes for message command file (INMSG) syntax errors.

2014 ALIAS invalid on LIST command.

Explanation: You specified an invalid value for an ALIAS parameter in the LIST command. The first character of the ALIAS is the destination table type and must be blank, g, o, or p. The last three characters of the ALIAS are the destination table ID and must be one to three alphanumeric characters. If the first character is blank, then the last three characters of the ALIAS parameter must be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the LIST command in the message command file, INMSG, and retry the program.

2032 LISTNAME missing on LIST command.

Explanation: The LISTNAME parameter in the LIST command is missing or blank. A LISTNAME must be specified.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which LIST command produced the error. Correct the LIST command in the message command file, INMSG, and retry the program.

2062 FUNCTION missing or invalid on LIST command.

Explanation: The FUNCTION parameter in the LIST command is missing, or you have specified an invalid value. The FUNCTION value must be a, d, e, or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which LIST command produced the error. Correct the LIST command in the message command file, INMSG, and retry the program.

2064 Invalid entries on LIST command.

Explanation: Either entries were specified in the LIST command with an 'E' for FUNCTION parameter, or no entries were specified for one of the other FUNCTION values.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which LIST command produced the error. Correct the LIST command in the message command file, INMSG, and retry the program.

2066 LISTTYPE invalid on LIST command.

Explanation: The LISTTYPE value in the LIST command is invalid. The LISTTYPE value must be t, p, a, or g.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the LIST command in the message command file, INMSG, and retry the program.

2098 Invalid command sequence on LIST command.

Explanation: A LISTNAME, FUNCTION, or LISTTYPE parameter was specified more than once on the LIST command. You can only specify these parameters once in each LIST command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the LIST command in the message command file, INMSG, and retry the program.

2099 Incomplete destination on LIST command.

Explanation: The list ended or a new destination was started before a previous destination was finished. This is usually caused by a missing parameter somewhere in the list. List entries are made up of either an ACCOUNT and USERID, or an ALIAS and ALIASNAME. Therefore, you must specify each list entry component next to its counterpart. For example, an ACCOUNT parameter should be entered next to a USERID parameter. If you specify the SYSID parameter, you must specify it either before the ACCOUNT and USERID parameters to which it belongs or between them. It cannot follow the ACCOUNT and USERID parameters for that SYSID.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the LIST command in the message command file, INMSG, and retry the program.

2201 Both KEYRINGSTASHFILE and KEYRINGPASSWORD specified on START command.

Explanation: You specified both the KEYRINGSTASHFILE and the KEYRINGPASSWORD parameters on the START command.

User Response: When the file specified in the KEYRINGFILE parameter was created, it was either created to have a password or to store its password in a stash file. If the password was stored in a stash file, specify the name of the stash file in the KEYRINGSTASHFILE parameter. Otherwise, specify the password in the KEYRINGPASSWORD parameter. Correct the START command in the message command file, INMSG, and retry the program.

2202 KEYRINGFILE missing on START command.

Explanation: You must specify the KEYRINGFILE parameter on the START command.

User Response: When making an SSL connection, there must be a KDB file containing the certificate to allow the connection. This file name is specified in the KEYRINGFILE parameter. This parameter is required when SSL is enabled. Correct the START command in the message command file, INMSG to specify the filename of the KEYRINGFILE parameter, and retry the program.

2203 KEYRINGPASSWORD missing on START command.

Explanation: The key database or key ring file specified in the KEYRINGFILE parameter on the START command requires a password.

User Response: Provide either the correct KEYRINGPASSWORD parameter or the correct KEYRINGSTASHFILE associated with the key database specified in the KEYRINGFILE parameter, and retry the program.

2204 KEYRINGPASSWORD or KEYRINGSTASHFILE not allowed on START command.

Explanation: When using a RACF key ring file, the KEYRINGPASSWORD and KEYRINGSTASHFILE parameters cannot be specified.

User Response: Remove the KEYRINGPASSWORD or KEYRINGSTASHFILE parameter and retry the program.

2402 No destination on CANCEL command.

Explanation: You must specify a destination for this command. You can use ACCOUNT and USERID; ALIAS and ALIASNAME; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2404 Multiple destinations on CANCEL command.

Explanation: You can specify only one destination in the CANCEL command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2406 Insufficient destination on CANCEL command.

Explanation: You must specify a complete destination for this command. You can use ACCOUNT and USERID; ALIAS and ALIASNAME; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2414 ALIAS invalid on CANCEL command.

Explanation: You specified an invalid value for the ALIAS parameter in the CANCEL command. The first character of the ALIAS parameter is the destination table type and must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID and must be one to three alphanumeric characters. If the first character is blank, then the last three characters of the ALIAS parameter must be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2444 PRIORITY invalid on CANCEL command.

Explanation: You specified an invalid value for the PRIORITY parameter in the CANCEL command. The value of the PRIORITY parameter must be blank or p.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2450 ACK invalid on CANCEL command.

Explanation: You specified an invalid value for the ACK parameter in the CANCEL command. The value of the ACK parameter must be blank, h, or t.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2464 STARTDATE invalid on CANCEL command.

Explanation: You specified an invalid value for the STARTDATE parameter in the CANCEL command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2466 STARTTIME invalid on CANCEL command.

Explanation: You specified an invalid value for the STARTTIME parameter in the CANCEL command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2468 ENDDATE invalid on CANCEL command.

Explanation: You specified an invalid value for the ENDDATE parameter in the CANCEL command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2470 ENDTIME invalid on CANCEL command.

Explanation: You specified an invalid value for the ENDTIME parameter in the CANCEL command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2472 TIMEZONE invalid on CANCEL command.

Explanation: You specified an invalid value for the TIMEZONE parameter in the CANCEL command. TIMEZONE must be either g or l.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2474 End date/time is before start date/time.

Explanation: You specified an end date and time in the CANCEL command that is before the start date and time. If you specified either date field with a two-digit year, a sliding window technique was used to decide the century indicator that is used when comparing the start and end dates. The two-digit year will be considered a 20xx year if the two digits are between 00 and the current year + 49. For example:

- In 1997, two-digit years starting with 00 through 46 will be considered 20xx years, while two-digit years starting with 47 through 99 will be considered 19xx years.
- In 1998, two-digit years starting with 00 through 47 will be considered 20xx years, while two-digit years starting with 48 through 99 will be considered 19xx years.
- In 1999, two-digit years starting with 00 through 48 will be considered 20xx years, while two-digit years starting with 49 through 99 will be considered 19xx years.
- In 2000, two-digit years starting with 00 through 49 will be considered 20xx years, while two-digit years starting with 50 through 99 will be considered 19xx years.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which CANCEL command produced the error. Correct the CANCEL command in the message command file, INMSG, and retry the program.

2604 Multiple destinations on AUDIT command.

Explanation: You can specify only one destination in the AUDIT command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2606 Incomplete destination on AUDIT command.

Explanation: You specified an incomplete destination in the AUDIT command. If you specify a destination, you must use ACCOUNT and USERID; SYSID, ACCOUNT and USERID; or ALIAS and ALIASNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2614 ALIAS invalid on AUDIT command.

Explanation: You specified an invalid value for the ALIAS parameter in the AUDIT command. The first character of the ALIAS parameter is the destination table type and must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID and must be one to three alphanumeric characters. If the first character is blank, then the last three characters of the ALIAS parameter must be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2664 STARTDATE invalid on AUDIT command.

Explanation: You specified an invalid value for the STARTDATE parameter in the AUDIT command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2668 ENDDATE invalid on AUDIT command.

Explanation: You specified an invalid value for the ENDDATE parameter in the AUDIT command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2670 ALTACCT invalid on AUDIT command.

Explanation: You specified ALTACCT and did not specify ALTUSERID or you specified ALTACCT and LEVEL 1 or 2. If you specify ALTACCT you must specify ALTUSERID. ALTACCT is only supported for LEVEL 3 or higher.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2672 TIMEZONE invalid on AUDIT command.

Explanation: You specified an invalid value in the TIMEZONE parameter on the AUDIT command. TIMEZONE must be either g or l.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2674 STATUS invalid on AUDIT command.

Explanation: You specified an invalid value for the STATUS parameter in the AUDIT command. STATUS must be blank, u, p, or d.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2676 MSGTYPE invalid on AUDIT command.

Explanation: You specified an invalid value for the MSGTYPE parameter in the AUDIT command. MSGTYPE must be s, r, or b.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2678 End date is before start date.

Explanation: You specified an end date in the AUDIT command that is before the start date. If you specified either date field as a two-digit year, a sliding window technique was used to decide the century indicator that is used when comparing the start and end dates. The two-digit year will be considered a 20xx year if the two digits are between 00 and the current year + 49. For example:

- In 1997, two-digit years starting with 00 through 46 will be considered 20xx years, while two-digit years starting with 47 through 99 will be considered 19xx years.
- In 1998, two-digit years starting with 00 through 47 will be considered 20xx years, while two-digit years starting with 48 through 99 will be considered 19xx years.
- In 1999, two-digit years starting with 00 through 48 will be considered 20xx years, while two-digit years starting with 49 through 99 will be considered 19xx years.
- In 2000, two-digit years starting with 00 through 49 will be considered 20xx years, while two-digit years starting with 50 through 99 will be considered 19xx years.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2684 LEVEL invalid on AUDIT command.

Explanation: You specified an invalid value for the LEVEL parameter in the AUDIT command. LEVEL must be 1, 2, or 3.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which AUDIT command produced the error. Correct the AUDIT command in the message command file, INMSG, and retry the program.

2802 No destination specified on SEND command.

Explanation: You must specify a destination in the SEND command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2804 Multiple destinations on SEND command.

Explanation: You specified multiple destinations in a SEND command. You can only specify one destination.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2806 Incomplete destination on SEND command.

Explanation: You specified an incomplete destination in the SEND command. The destination must be an ACCOUNT and USERID; SYSID, ACCOUNT and USERID; ALIAS and ALIASNAME; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2814 ALIAS invalid on SEND command.

Explanation: You specified an invalid value for the ALIAS parameter in the SEND command. The first character of the ALIAS parameter is the destination table type and must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID and must be one to three alphanumeric characters. If the first character is blank, then the last three characters of the ALIAS parameter must be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2830 No FILEID specified on SEND command.

Explanation: You must specify a FILEID in the SEND command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2836 FORMAT invalid on SEND command.

Explanation: You specified an invalid value for the FORMAT parameter in the SEND command. FORMAT must be y or n. You cannot specify 'Y' for FORMAT with 'B' for DATATYPE or 'C' or ‘L’ for DELIMIT.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2844 PRIORITY invalid on SEND command.

Explanation: You specified an invalid value for the PRIORITY parameter in the SEND command. PRIORITY must be blank, i, or p.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2846 MODE invalid on SEND command.

Explanation: You specified an invalid value for the MODE parameter in the SEND command. MODE must be blank or t.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2848 CHARGE invalid on SEND command.

Explanation: You specified an invalid value for the CHARGE parameter in the SEND command. CHARGE must be a numeric character from 1 to 6.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2850 ACK invalid on SEND command.

Explanation: You specified an invalid value for the ACK parameter in the SEND command. ACK must be blank, a, b, c, d, e, f, or r.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2872 VERIFY invalid on SEND command.

Explanation: You specified an invalid value for the VERIFY parameter in the SEND command. VERIFY must be y, n, or f.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2874 RETAIN invalid on SEND command.

Explanation: You specified an invalid value for the RETAIN parameter in the SEND command. RETAIN must be a numeric value from 0 to 180.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2876 DATATYPE invalid on SEND command.

Explanation: You specified an invalid value for the DATATYPE parameter in the SEND command. DATATYPE must be e or b.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2878 DELIMIT invalid on SEND command.

Explanation: You specified an invalid value for the DELIMIT parameter in the SEND command. DELIMIT must be c, l, u, or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2890 CRLFEOF invalid on SEND command.

Explanation: You specified an invalid value for the CRLFEOF parameter in the SEND command. CRLFEOF must contain 6 hexadecimal characters.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2892 TRUNCATE invalid on SEND command.

Explanation: You specified an invalid value for the TRUNCATE parameter in the SEND command. TRUNCATE must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2898 COMPRESS invalid on SEND command.

Explanation: You specified an invalid value for the COMPRESS parameter in the SEND command. COMPRESS must be y, n, or t.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2899 SELECTRCV invalid on SEND command.

Explanation: You specified an invalid value for the SELECTRCV parameter in the SEND command. SELECTRCV must be f, n, or blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SEND command produced the error. Correct the SEND command in the message command file, INMSG, and retry the program.

2902 No destination specified on SENDSTREAM command.

Explanation: You must specify a destination in the SENDSTREAM command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2904 Multiple destinations on SENDSTREAM command.

Explanation: You specified multiple destinations in the SENDSTREAM command. You can specify only one destination.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2906 Incomplete destination on SENDSTREAM command.

Explanation: You specified an incomplete destination in the SENDSTREAM command. The destination must be ALIAS and ALIASNAME; ACCOUNT and USERID; SYSID with ACCOUNT and USERID; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2914 ALIAS invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the ALIAS parameter in the SENDSTREAM command. The first character of the ALIAS parameter is the destination table type, and it must be blank, g, o, or p. The last three characters of the ALIAS parameter indicate the destination table, and they must be 1 to 3 alphanumeric characters.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2944 PRIORITY invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the PRIORITY parameter in the SENDSTREAM command. PRIORITY must be blank, i, or p.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2946 MODE invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the MODE parameter in the SENDSTREAM command. MODE must be blank or t.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2948 CHARGE invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the CHARGE parameter in the SENDSTREAM command. CHARGE must be a numeric character from 1 to 6.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2950 ACK invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the ACK parameter in the SENDSTREAM command. ACK must be blank, a, b, c, d, e, f, or r.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2964 No ENDSTR parameter specified on SENDSTREAM command.

Explanation: You must specify a value for the ENDSTR parameter in the SENDSTREAM command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2972 VERIFY invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the VERIFY parameter in the SENDSTREAM command. VERIFY must be y, n, or f.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2974 RETAIN invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the RETAIN parameter in the SENDSTREAM command. RETAIN must be a numeric value from 0 to 180.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2976 DATATYPE invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the DATATYPE parameter in the SENDSTREAM command. DATATYPE must be e or b.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2993 EDITYPE invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the EDITYPE parameter in the SENDSTREAM command. EDITYPE must be n, e, t, u, or x.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2994 SENDSTREAM command is only valid for session-level recovery.

Explanation: The SENDSTREAM command cannot be used for checkpoint-level, file-level or user-initiated recovery. It is only valid with session-level recovery.

User Response: Either use a SEND instead of the SENDSTREAM command, and put data in a separate data set, or change the RECOVERY parameter in the profile to use session-level recovery.

2996 End of data for SENDSTREAM command not found.

Explanation: The value specified by the ENDSTR parameter was not found in the data before the end of the file was reached. This value must be at the end of the data to indicate where the data ends.

User Response: Check the command to be sure that the ENDSTR value is correct and that it appears at the end of the data exactly as it appears in the parameter value. Note that the search for the ENDSTR value is sensitive to uppercase and lowercase differences. Correct the command file and try again.

2998 COMPRESS invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the COMPRESS parameter in the SENDSTREAM command. COMPRESS must be y, n, or t.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

2999 SELECTRCV invalid on SENDSTREAM command.

Explanation: You specified an invalid value for the SELECTRCV parameter in the SENDSTREAM command. SELECTRCV must be f, n, or blank.

System Action: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDSTREAM command produced the error. Correct the SENDSTREAM command in the message command file, INMSG, and retry the program.

3030 No FILEID specified on SENDEDI command.

Explanation: You must specify a FILEID in the SENDEDI command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3044 PRIORITY invalid on SENDEDI command.

Explanation: You specified an invalid value for the PRIORITY parameter in the SENDEDI command. PRIORITY must be blank, i, or p.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3046 MODE invalid on SENDEDI command.

Explanation: You specified an invalid value for the MODE parameter in the SENDEDI command. MODE must be blank or t.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3048 CHARGE invalid on SENDEDI command.

Explanation: You specified an invalid value for the CHARGE parameter in the SENDEDI command. CHARGE must be a numeric character from 1 to 6.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3050 ACK invalid on SENDEDI command.

Explanation: You specified an invalid value for the ACK parameter in the SENDEDI command. ACK must be blank, a, b, c, d, e, f, or r.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3072 VERIFY invalid on SENDEDI command.

Explanation: You specified an invalid value for the VERIFY parameter in the SENDEDI command. VERIFY must be y, c, n, f, or g.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3074 RETAIN invalid on SENDEDI command.

Explanation: You specified an invalid value for the RETAIN parameter in the SENDEDI command. RETAIN must be a numeric value from 0 to 180.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3098 COMPRESS invalid on SENDEDI command.

Explanation: You specified an invalid value for the COMPRESS parameter in the SENDEDI command. COMPRESS must be y, n, or t.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3099 SELECTRCV invalid on SENDEDI command.

Explanation: You specified an invalid value for the SELECTRCV parameter in the SENDEDI command. COMPRESS must be f, n, or blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which SENDEDI command produced the error. Correct the SENDEDI command in the message command file, INMSG, and retry the program.

3204 Multiple sources on RECEIVE command.

Explanation: You specified multiple sources in the RECEIVE command. You can specify only one source.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3206 Incomplete source specified on RECEIVE command.

Explanation: You specified an incomplete source for the RECEIVE command. If you specify a source, you must use ACCOUNT and USERID; SYSID, ACCOUNT and USERID; ALIAS and ALIASNAME; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3214 ALIAS invalid on RECEIVE command.

Explanation: You specified an invalid value for the ALIAS parameter in the RECEIVE command. The first character of the ALIAS parameter is the destination table type, and it must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID, and they must be 1 to 3 alphanumeric characters. If the first character is blank, the last three characters of the ALIAS parameter must also be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3230 No FILEID specified on RECEIVE command.

Explanation: You must specify a FILEID in the RECEIVE command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3236 FORMAT invalid on RECEIVE command.

Explanation: You specified an invalid value for the FORMAT parameter in the RECEIVE command. FORMAT must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3242 ALLFILES invalid on RECEIVE command.

Explanation: You specified an invalid value for the ALLFILES parameter in the RECEIVE command. ALLFILES must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3246 EDIOPT invalid on RECEIVE command.

Explanation: You specified an invalid value for the EDIOPT parameter in the RECEIVE command. EDIOPT must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3256 REQUEUED invalid on RECEIVE command.

Explanation: You specified an invalid value for the REQUEUED parameter in the RECEIVE command. REQUEUED must be y or n. If you specify a source, REQUEUED cannot be y.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3258 AUTOEDI invalid on RECEIVE command.

Explanation: You specified an invalid value for the AUTOEDI parameter in the RECEIVE command. AUTOEDI must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3260 DELIMITED invalid on RECEIVE command.

Explanation: You specified an invalid value for the DELIMITED parameter in the RECEIVE command. DELIMITED must be c, l, or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3262 DLMOVERRIDE invalid on RECEIVE command.

Explanation: You specified an invalid value for the DLMOVERRIDE parameter in the RECEIVE command. DLMOVERRIDE must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3264 ENDSTR invalid on RECEIVE command.

Explanation: You specified an invalid value for the ENDSTR parameter in the RECEIVE command. The ENDSTR parameter must be 1 to 79 characters. If it includes an s: or r: to indicate whether it starts on a new record, these characters can be added to the 79 characters for a maximum length of 81.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3266 MSGKEY invalid on RECEIVE command.

Explanation: You specified an invalid value for the MSGKEY parameter in the RECEIVE command. MSGKEY must be 20 hex characters.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3270 STARTDATE invalid on RECEIVE command.

Explanation: You specified an invalid value for the STARTDATE parameter in the RECEIVE command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3271 STARTTIME invalid on RECEIVE command.

Explanation: You specified an invalid value for the STARTTIME parameter in the RECEIVE command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3272 ENDDATE invalid on RECEIVE command.

Explanation: You specified an invalid value for the ENDDATE parameter in the RECEIVE command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3273 ENDTIME invalid on RECEIVE command.

Explanation: You specified an invalid value for the ENDTTIME parameter in the RECEIVE command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3274 TIMEZONE invalid on RECEIVE command.

Explanation: You specified an invalid value for the TIMEZONE parameter in the RECEIVE command. TIMEZONE must be g or l.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3275 End date/time is before start date/time.

Explanation: You specified an end date and time in the RECEIVE command that is before the start date and time. If you specified either date field with a two-digit year, a sliding window technique was used to decide the century indicator that is used when comparing the start and end dates. The two-digit year will be considered a 20xx year if the two digits are between 00 and the current year + 49. For example:

- In 1997, two-digit years starting with 00 through 46 will be considered 20xx years, while two-digit years starting with 47 through 99 will be considered 19xx years
- In 1998, two-digit years starting with 00 through 47 will be considered 20xx years, while two-digit years starting with 48 through 99 will be considered 19xx years.
- In 1999, two-digit years starting with 00 through 48 will be considered 20xx years, while two-digit years starting with 49 through 99 will be considered 19xx years.
- In 2000, two-digit years starting with 00 through 49 will be considered 20xx years, while two-digit years starting with 50 through 99 will be considered 19xx years.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3276 RESRECL invalid on RECEIVE command.

Explanation: You specified an invalid value for the RESRECL parameter in the RECEIVE command. RESRECL must be s or e. If you are using checkpoint-level recovery, only s is valid.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3284 NONEDIONLY invalid on RECEIVE command.

Explanation: You specified an invalid value for the NONEDIONLY parameter in the RECEIVE command. NONEDIONLY must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVE command produced the error. Correct the RECEIVE command in the message command file, INMSG, and retry the program.

3304 Multiple sources on RECEIVESTREAM command.

Explanation: You specified multiple sources in the RECEIVESTREAM command. You can specify only one source.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3306 Incomplete source specified on RECEIVESTREAM command.

Explanation: You specified an incomplete source for this command. If you specify a source, you must use the parameters ACCOUNT and USERID; SYSID, ACCOUNT and USERID; ALIAS and ALIASNAME; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3314 ALIAS invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the ALIAS parameter in the RECEIVESTREAM command. The first character of the ALIAS parameter is the destination table type, and it must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID and must be 1 to 3 alphanumeric characters. If the first character is blank, the last three characters of the ALIAS parameter must also be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3342 ALLFILES invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the ALLFILES parameter in the RECEIVESTREAM command. ALLFILES must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3356 REQUEUED invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the REQUEUED parameter in the RECEIVESTREAM command. REQUEUED must be y or n. If you specify a source, REQUEUED cannot be y.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3364 ENDSTR missing or invalid on RECEIVESTREAM command.

Explanation: You either did not specify the ENDSTR parameter in the RECEIVESTREAM command, or you specified an invalid value. The ENDSTR parameter must be 1 to 79 characters. If it includes s: or r: to indicate whether it starts on a new record, these characters can be added to the 79 characters for a maximum length of 81.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3366 MSGKEY invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the MSGKEY parameter in the RECEIVESTREAM command. MSGKEY must either be blank or contain 20 hexadecimal characters.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3370 STARTDATE invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the STARTDATE parameter in the RECEIVESTREAM command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3371 STARTTIME invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the STARTTIME parameter in the RECEIVESTREAM command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3372 ENDDATE invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the ENDDATE parameter in the RECEIVESTREAM command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3373 ENDTIME invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the ENDTTIME parameter in the RECEIVESTREAM command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3374 TIMEZONE invalid on RECEIVESTREAM command.

Explanation: You specified an invalid value for the TIMEZONE parameter in the RECEIVESTREAM command. TIMEZONE must be either g or l.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3375 End date/time is before start date/time.

Explanation: You specified an end date and time in the RECEIVESTREAM command that is before the start date and time. If you specified either date field with a two-digit year, a sliding window technique was used to decide the century indicator that is used when comparing the start and end dates. The two-digit year will be considered a 20xx year if the two digits are between 00 and the current year + 49. For example:

- In 1997, two-digit years starting with 00 through 46 will be considered 20xx years, while two-digit years starting with 47 through 99 will be considered 19xx years.
- In 1998, two-digit years starting with 00 through 47 will be considered 20xx years, while two-digit years starting with 48 through 99 will be considered 19xx years.
- In 1999, two-digit years starting with 00 through 48 will be considered 20xx years, while two-digit years starting with 49 through 99 will be considered 19xx years.
- In 2000, two-digit years starting with 00 through 49 will be considered 20xx years, while two-digit years starting with 50 through 99 will be considered 19xx years.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVESTREAM command produced the error. Correct the RECEIVESTREAM command in the message command file, INMSG, and retry the program.

3394 RECEIVESTREAM command is only valid for session-level recovery.

Explanation: The RECEIVESTREAM command cannot be used for checkpoint-level, file-level, or user-initiated recovery. It is only valid with session-level recovery.

User Response: Either use a RECEIVE command instead of the RECEIVESTREAM command, and receive data into a separate data set, or change the RECOVERY parameter in the profile to use session-level recovery.

3404 Multiple sources in RECEIVEEDI command.

Explanation: You specified multiple sources in the RECEIVEEDI command. You can specify only one source.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3406 Incomplete source on RECEIVEEDI command.

Explanation: You specified an incomplete source in the RECEIVEEDI command. If you specify a source, you must use ACCOUNT and USERID; SYSID, ACCOUNT and USERID; ALIAS and ALIASNAME; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3414 ALIAS invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the ALIAS parameter in the RECEIVEEDI command. The first character of the ALIAS parameter is the destination table type, and it must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID, and they must be one to three alphanumeric characters. If the first character is blank, then the last three characters of the ALIAS parameter must be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3430 No FILEID specified on RECEIVEEDI command.

Explanation: You must specify a FILEID parameter in the RECEIVEEDI command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3442 ALLFILES invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the ALLFILES parameter in the RECEIVEEDI command. ALLFILES must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3446 EDIOPT invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the EDIOPT parameter in the RECEIVEEDI command. EDIOPT must be y, n, or f.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3456 REQUEUED invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the REQUEUED parameter in the RECEIVEEDI command. REQUEUED must be y or n. If you specify a source, REQUEUED cannot be y.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3466 MSGKEY invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the MSGKEY parameter in the RECEIVEEDI command. MSGKEY must be 20 hex characters.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the message command file, INMSG, and retry the program.

3470 STARTDATE invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the STARTDATE parameter in the RECEIVEEDI command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3471 STARTTIME invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the STARTTIME parameter in the RECEIVEEDI command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3472 ENDDATE invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the ENDDATE parameter in the RECEIVEEDI command. The format must be YYMMDD, where YY is the last two digits of the year, MM is the two-digit month, and DD is the two-digit day; or YYYYMMDD, where YYYY is a four-digit year.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3473 ENDTIME invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the ENDTTIME parameter in the RECEIVEEDI command. The format must be HHMMSS, where HH is the two-digit hour, MM is the two-digit minutes, and SS is the two-digit seconds.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3474 TIMEZONE invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the TIMEZONE parameter in the RECEIVEEDI command. TIMEZONE must be either g or l.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3475 End date/time is before start date/time.

Explanation: You specified an end date and time in the RECEIVEEDI command that is before the start date and time. If you specified either date field with a two-digit year, a sliding window technique was used to decide the century indicator that is used when comparing the start and end dates. The two-digit year will be considered a 20xx year if the two digits are between 00 and the current year + 49. For example:

- In 1997, two-digit years starting with 00 through 46 will be considered 20xx years, while two-digit years starting with 47 through 99 will be considered 19xx years.
- In 1998, two-digit years starting with 00 through 47 will be considered 20xx years, while two-digit years starting with 48 through 99 will be considered 19xx years.
- In 1999, two-digit years starting with 00 through 48 will be considered 20xx years, while two-digit years starting with 49 through 99 will be considered 19xx years.
- In 2000, two-digit years starting with 00 through 49 will be considered 20xx years, while two-digit years starting with 50 through 99 will be considered 19xx years.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3484 EDIONLY invalid on RECEIVEEDI command.

Explanation: You specified an invalid value for the EDIONLY parameter in the RECEIVEEDI command. EDIONLY must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which RECEIVEEDI command produced the error. Correct the RECEIVEEDI command in the message command file, INMSG, and retry the program.

3600 Requested a session start when already in session.

Explanation: You requested that Expedite Base do a session start when it was already in an Information Exchange session. If you use multiple START commands in a command file, you must end the previous Information Exchange session before you start the next one.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the message command file, INMSG, and retry the program.

3610 START command invalid with automatic session start.

Explanation: You specified the START command in the command file, but your profile, INPRO, indicates that the system should start the session automatically. If the session starts automatically, the START command is invalid.

User Response: Specify 'N' for AUTOSTART on the TRANSMIT command in the profile command file, INPRO, if you do not want the session started automatically. Otherwise, remove the START command from the message command file, INMSG, and retry the program.

3620 END command invalid with automatic session end.

Explanation: You specified the END command in the command file, but your profile indicates that the session should end automatically. If the system ends the session automatically, the END command is invalid.

User Response: Specify 'N' for AUTOEND on the TRANSMIT command in the profile command file, INPRO, if you do not want the session ended automatically. Otherwise, remove the END command from the message command file, INMSG, and retry the program.

3630 Session not started before first command.

Explanation: You specified a command in the message command file, INMSG, other than START, but were not currently in session.

User Response: Add a START command to the message command file, INMSG, or specify 'Y' for AUTOSTART on the TRANSMIT command in the profile command file, INPRO, and retry the program.

3640 No END command or automatic end for session.

Explanation: If you specify 'N' for AUTOEND on the TRANSMIT command, you must specify an END command in the command file.

User Response: Add an END command in the message command file, INMSG, or specify 'Y' for AUTOEND on the TRANSMIT command in the profile command file, INPRO. Retry the program.

3650 IE account/userid already in use.

Explanation: The account ID and user ID specified are already being used for another session.

User Response: When using checkpoint-level recovery, only one job should use an Information Exchange account ID/user ID combination at a time. The first job should be completed with a valid session end before any other jobs are submitted using that Information Exchange account ID and user ID combination.

3860 CDH invalid on QUERY command.

Explanation: You specified an invalid value for the CDH parameter in the QUERY command. CDH must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which QUERY command produced the error. Correct the QUERY command in the message command file, INMSG, and retry the program.

4060 Missing ARCHIVEID on ARCHIVEMOVE command.

Explanation: You did not specify an ARCHIVEID value on the ARCHIVEMOVE command. This is a required parameter.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which ARCHIVEMOVE command produced the error. Correct the ARCHIVEMOVE command in the message command file, INMSG, and retry the program.

4110 MSGKEY missing on PURGE command.

Explanation: You did not specify the MSGKEY parameter on the PURGE command. This is a required parameter.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PURGE command produced the error. Correct the PURGE command in the message command file, INMSG, and retry the program. Valid message keys can be found on the AVAILABLE record in response to a QUERY command.

4232 LISTNAME missing or invalid

Explanation: The LISTNAME value on the LISTVERIFY command must be specified if the FUNCTION value is b, c, l, r, or s, but cannot be specified if the FUNCTION value is a or d.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which LISTVERIFY command produced the error. Correct the LISTVERIFY command in the message command file, INMSG, and retry the program.

4248 Invalid CHARGE on LISTVERIFY command.

Explanation: You specified an invalid value for the CHARGE parameter on the LISTVERIFY command. CHARGE must be a numeric character from 1 to 6.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which LISTVERIFY command produced the error. Correct the LISTVERIFY command in the message command file, INMSG, and retry the program.

4266 FUNCTION missing or invalid on LISTVERIFY command.

Explanation: The FUNCTION value on the LISTVERIFY command is invalid or was not specified. A FUNCTION value is required and must be a, b, c, d, l, r, or s.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which LISTVERIFY command produced the error. Correct the LISTVERIFY command in the message command file, INMSG, and retry the program.

4460 STARTMSG missing or invalid on TESTMSG command.

Explanation: The STARTMSG value on the TESTMSG command is invalid or was not specified. A STARTMSG value is required and must be a numeric value from 0 to 5.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which TESTMSG command produced the error. Correct the TESTMSG command in the message command file, INMSG, and retry the program.

4462 ENDMSG missing or invalid on TESTMSG command.

Explanation: The ENDMSG value on the TESTMSG command is invalid or was not specified. An ENDMSG value is required. It must be a numeric value from 0 to 5 and be greater than or equal to the STARTMSG value.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which TESTMSG command produced the error. Correct the TESTMSG command in the message command file, INMSG, and retry the program.

4502 Missing destination on DEFINEALIAS command.

Explanation: You did not specify a destination on the DEFINEALIAS command. A destination consists of an ACCOUNT and USERID; SYSID, ACCOUNT and USERID; or ALIAS and ALIASNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4514 Invalid table type on DEFINEALIAS command.

Explanation: You specified an invalid value for an ALIAS parameter in the DEFINEALIAS command. The first character of the ALIAS parameter is the destination table type, and it must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID, and they must be 1 to 3 alphanumeric characters. If the first character is blank, then the last three characters of the ALIAS parameter must be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4532 Invalid table name on DEFINEALIAS command.

Explanation: The ALIASTABLE parameter in the DEFINEALIAS command is missing, blank, or invalid. An ALIASTABLE name must be specified. The first character of the ALIASTABLE parameter is the destination table type, and it must be g, o, or p. The last three characters of the ALIAS parameter are the destination table ID, and they must be 1 to 3 alphanumeric characters.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4562 FUNCTION invalid on DEFINEALIAS command.

Explanation: The value for the FUNCTION parameter in the DEFINEALIAS command is invalid. The FUNCTION parameter value must be a, c, d, e, or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4564 Invalid use of FUNCTION parameter on DEFINEALIAS command.

Explanation: Either entries were specified in the DEFINEALIAS command with 'E' for FUNCTION, or no entries were specified for one of the other FUNCTION values.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4566 AUTHORITY invalid on DEFINEALIAS command.

Explanation: Use of the AUTHORITY parameter in the DEFINEALIAS command is invalid or you specified an invalid value for the AUTHORITY parameter. You may use the AUTHORITY parameter only if you are defining a new alias table, when FUNCTION is set to 'N'. Valid values for the AUTHORITY parameter are p, a, or g. If you are defining a private alias table, where the first character of the alias table name is P, or an organizational alias table, where the first character of the alias table name is O, then you cannot specify 'G' for AUTHORITY.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4598 Invalid parameter duplication on DEFINEALIAS command.

Explanation: An ALIASTABLE, FUNCTION, or AUTHORITY parameter is duplicated. These parameters can be specified only once in each DEFINEALIAS command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4599 Incomplete destination on DEFINEALIAS command.

Explanation: A DEFINEALIAS command ended before a destination was completed, or a new type of destination started before the previous destination entry was completed. This is usually caused by a missing parameter somewhere in a destination of a DEFINEALIAS command. Destination entries are made up of either an ACCOUNT and USERID; SYSID, ACCOUNT and USERID; or an ALIAS and ALIASNAME. Therefore, you must specify each destination component next to its counterpart. For example, an ACCOUNT parameter should be entered next to a USERID parameter. If you specify a SYSID parameter, you must specify it next to its associated ACCOUNT and USERID parameters. An ALIAS parameter must have an associated ALIASNAME parameter. A SYSID next to an ALIASNAME is invalid.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which DEFINEALIAS command produced the error. Correct the DEFINEALIAS command in the message command file, INMSG, and retry the program.

4630 FILEID missing on PUTMEMBER command.

Explanation: You must specify a FILEID in the PUTMEMBER command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4636 FORMAT invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the FORMAT parameter in the PUTMEMBER command. FORMAT must be y or n. You cannot specify 'Y' for FORMAT with 'B' for DATATYPE or 'C' or ‘L’ for DELIMIT.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4650 ACK invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the ACK parameter in the PUTMEMBER command. ACK must be blank or d.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4672 VERIFY invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the VERIFY parameter in the PUTMEMBER command. VERIFY must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4676 DATATYPE invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the DATATYPE parameter in the PUTMEMBER command. DATATYPE must be e or b.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4678 DELIMIT invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the DELIMIT parameter in the PUTMEMBER command. DELIMIT must be c, l, or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4686 MEMBER missing on PUTMEMBER command.

Explanation: You must specify a MEMBER name on the PUTMEMBER command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4688 LIBRARY missing on PUTMEMBER command.

Explanation: You must specify a LIBRARY name on the PUTMEMBER command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4690 CRLFEOF invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the CRLFEOF parameter in the PUTMEMBER command. CRLFEOF must contain 6 hexadecimal characters.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4692 TRUNCATE invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the TRUNCATE parameter in the PUTMEMBER command. TRUNCATE parameter must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4694 REPLACE invalid on PUTMEMBER command.

Explanation: You specified an invalid value for the REPLACE parameter in the PUTMEMBER command. REPLACE must be y or n.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which PUTMEMBER command produced the error. Correct the PUTMEMBER command in the message command file, INMSG, and retry the program.

4704 Multiple destinations on GETMEMBER command.

Explanation: You specified multiple destinations in a GETMEMBER command. You can specify only one destination.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4706 Partial destinations on GETMEMBER command.

Explanation: The destination must be ALIAS and ALIASNAME; ACCOUNT and USERID; SYSID, ACCOUNT and USERID; or LISTNAME.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4714 Invalid ALIAS on GETMEMBER command.

Explanation: You specified an invalid value for an ALIAS parameter in the GETMEMBER command. The first character of the ALIAS parameter is the destination table type, and it must be blank, g, o, or p. The last three characters of the ALIAS parameter are the destination table ID, and they must be 1 to 3 alphanumeric characters. If the first character is blank, the last three characters of the ALIAS parameter must also be blank.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4748 CHARGE invalid on GETMEMBER command.

Explanation: You specified an invalid value for the CHARGE parameter in the GETMEMBER command. CHARGE must be one of the following: 1, 3, 5, or 6.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4750 ACK invalid on GETMEMBER command.

Explanation: You specified an invalid value for the ACK parameter in the GETMEMBER command. ACK must be blank, a, b, c, d, e, f, or r.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4774 RETAIN invalid on GETMEMBER command.

Explanation: You specified an invalid value for the RETAIN parameter in the GETMEMBER command. RETAIN must be a numeric value from 0 to 180.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4786 MEMBER name missing on GETMEMBER command.

Explanation: You must specify a MEMBER name on the GETMEMBER command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4788 LIBRARY name missing on GETMEMBER command.

Explanation: You must specify a LIBRARY name on the GETMEMBER command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which GETMEMBER command produced the error. Correct the GETMEMBER command in the message command file, INMSG, and retry the program.

4866 AUTHORITY invalid on LISTLIBRARIES command.

Explanation: You specified an invalid value for the AUTHORITY parameter in the LISTLIBRARIES command. AUTHORITY must be r or w.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the message command file, INMSG, and try again.

4868 SELECTION invalid on LISTLIBRARIES command.

Explanation: You specified an invalid value for the SELECTION parameter in the LISTLIBRARIES command. SELECTION must be a or c.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which command produced the error. Correct the message command file, INMSG, and try again.

4988 LIBRARY name missing on LISTMEMBERS command.

Explanation: You must specify a LIBRARY name on the LISTMEMBERS command.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which LISTMEMBERS command produced the error. Correct the LISTMEMBERS command in the message command file, INMSG, and retry the command.

Profile command file syntax errors

The following describes the return codes for profile command file (INPRO) syntax errors.

5024 IEACCOUNT missing on IDENTIFY command.

Explanation: You must specify an Information Exchange account in your profile when Expedite Base starts the session automatically.

User Response: Specify the account using the IEACCOUNT parameter in the IDENTIFY command in the profile command file, INPRO. Retry the program.

5026 IEUSERID missing on IDENTIFY command.

Explanation: You must specify an Information Exchange user ID in your profile when Expedite Base starts the session automatically.

User Response: Specify the user ID using the IEUSERID parameter in the IDENTIFY command in the profile command file, INPRO. Retry the program.

5028 IEPASSWORD missing on IDENTIFY command.

Explanation: You must specify an Information Exchange password in your profile when Expedite Base starts the session automatically.

User Response: Specify the password using the IEPASSWORD parameter in the IDENTIFY command in the profile command file, INPRO. Retry the program.

5029 Both KEYRINGSTASHFILE and KEYRINGPASSWORD on IDENTIFY command.

Explanation: You specified both the KEYRINGSTASHFILE and the KEYRINGPASSWORD parameters on the IDENTIFY command.

User Response: When the file specified in the KEYRINGFILE parameter was created, it was either created to have a password or to store its password in a stash file. If the password was stored in a stash file, specify the name of the stash file in the KEYRINGSTASHFILE parameter. Otherwise, specify the password in the KEYRINGPASSWORD parameter. Correct the IDENTIFY command to only specify the appropriate parameter, and retry the program.

5030 Missing KEYRINGFILE on IDENFITY command.

Explanation: The KEYRINGFILE was not specified. This is a required parameter. You must specify the name of the KEYRINGFILE.

User Response: When making an SSL connection, there must be a file containing the certificate to allow the connection. This filename is specified in the KEYRINGFILE parameter. It is a required parameter when SSL is enabled. Correct the IDENTIFY command and retry the program.

5031 IDENTIFY key ring file parameters not valid when AUTOSTART is N.

Explanation: The key ring parameters cannot be specified on the IDENTIFY command when AUTOSTART is set to n on the TRANSMIT command.

User Response: Remove all of the following parameters from the IDENTIFY command if they exist: KEYRINGFILE, KEYRINGPASSWORD, and KEYRINGSTASHFILE. As an alternative, you can set the AUTOSTART parameter to Y on the TRANSMIT command, remove the START command in INPRO, and retry the program.

5032 Missing KEYRINGPASSWORD on IDENTIFY command.

Explanation: The key database or key ring specified in the KEYRINGFILE parameter requires a password.

User Response: Provide either the correct KEYRINGPASSWORD parameter or KEYRINGSTASHFILE associated with the key database specified in the KEYRINGFILE parameter and retry the program.

5033 KEYRINGPASSWORD or KEYRINGSTASHFILE not allowed on IDENTIFY command.

Explanation: When using a RACF key ring file, the KEYRINGPASSWORD and KEYRINGSTASHFILE parameters cannot be specified.

User Response: Remove the KEYRINGPASSWORD or KEYRINGSTASHFILE parameter and retry the program.

5072 TIMEZONE invalid on IDENTIFY command.

Explanation: You specified an invalid value for the TIMEZONE parameter in the IDENTIFY command. TIMEZONE must be 1 to 5 alphanumeric characters. Valid values are ast, ahd, ahs, bst, cdt, cst, ead, edt, emt, est, gmt, jst, mdt, mst, pdt, pst, wed, wes, ydt, and yst. You can also specify the time zone as hours and minutes east or west of Greenwich mean time, for example, w0400 or e0400.

User Response: Correct the TIMEZONE parameter in the IDENTIFY command in the profile command file, INPRO. Retry the program.

5678 PROTOCOL invalid on TRACE command.

Explanation: You specified an invalid value for the PROTOCOL parameter in the TRACE command. PROTOCOL must be y or n.

User Response: Correct the PROTOCOL parameter in the TRACE command in the profile command file, INPRO. Retry the program.

5680 LINK invalid on TRACE command.

Explanation: You specified an invalid value for the LINK parameter in the TRACE command. LINK must be y or n.

User Response: Correct the LINK parameter in the TRACE command in the profile command file, INPRO. Retry the program.

5682 BASE invalid on TRACE command.

Explanation: You specified an invalid value for the BASE parameter in the TRACE command. BASE must be y or n.

User Response: Correct the BASE parameter in the TRACE command in the profile command file, INPRO. Retry the program.

5684 AUTOSTART invalid on TRANSMIT command.

Explanation: You specified an invalid value for the IOFILE parameter in the TRACE command. IOFILE must be y or n.

User Response: Correct the IOFILE parameter in the TRACE command in the profile command file, INPRO. Retry the program.

5872 AUTOSTART invalid on TRANSMIT command.

Explanation: You specified an invalid value for the AUTOSTART parameter in the TRANSMIT command. AUTOSTART must be y or n.

User Response: Correct the AUTOSTART parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

5876 AUTOEND invalid on TRANSMIT command.

Explanation: You specified an invalid value for the AUTOEND parameter in the TRANSMIT command. AUTOEND must be y or n.

User Response: Correct the AUTOEND parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

5878 COMMITDATA invalid on TRANSMIT command.

Explanation: You specified an invalid value for the COMMITDATA parameter in the TRANSMIT command. COMMITDATA must be a numeric value from 1000 to 37000 and must be greater than or equal to the MSGSIZE parameter value.

User Response: Correct the COMMITDATA parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

5882 COMMTYPE invalid on TRANSMIT command.

Explanation: You specified an invalid value for the COMMTYPE parameter in the TRANSMIT command. COMMTYPE must be s or t.

User Response: Correct the COMMTYPE parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

5884 MAXMSGS invalid on TRANSMIT command.

Explanation: You specified an invalid value for the MAXMSGS parameter in the TRANSMIT command. MXMSGS must be a numeric value from 1 to 9999.

User Response: Correct the MAXMSGS parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

5892 RECOVERY invalid on TRANSMIT command

Explanation: You specified an invalid value for the RECOVERY parameter in the TRANSMIT command. RECOVERY must be c, s, f, or u.

User Response: Correct the RECOVERY parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

5894 Session-level recovery not valid with session in progress.

Explanation: You specified session-level recovery, but the session file indicates that a checkpoint session is in progress.

User Response: Continue the present session using checkpoint-level recovery. If you want to use session-level recovery without completing the present session, either specify the RESET parameter on the IEBASE EXEC statement, or delete the session file, SESSION.

ATTENTION: If you reset the session using the RESET parameter, you will no longer be able to continue the previous session. Failure to modify the message command file, INMSG, before resetting the session may result in some data being lost or duplicated.

5896 MSGSIZE invalid on TRANSMIT command.

Explanation: You specified an invalid value for the MSGSIZE parameter in the TRANSMIT command. MSGSIZE must be a numeric value from 1000 to 37000.

User Response: Correct the MSGSIZE parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

5898 TIMEOUT invalid on TRANSMIT command.

Explanation: You specified an invalid value for the TIMEOUT parameter in the TRANSMIT command. TIMEOUT must be a number from 0 to 99.

User Response: Correct the TIMEOUT parameter in the TRANSMIT command in the profile command file, INPRO. Retry the program.

6010 USERLU missing on SNACOMM command.

Explanation: You must specify at least one USERLU value in the SNACOMM command.

User Response: Correct the USERLU parameter in the SNACOMM command in the profile command file, INPRO. Retry the program.

6012 Too many USERLU parameters on the SNACOMM command.

Explanation: You can specify a maximum of 99 USERLU parameters in the SNACOMM command.

User Response: Correct the USERLU parameter in the SNACOMM command in the profile command file, INPRO. Retry the program.

6014 RUSIZE invalid on SNACOMM command.

Explanation: You specified an invalid value for the RUSIZE parameter in the SNACOMM command. RUSIZE must be a numeric value from 256 through 3840.

User Response: Correct the RUSIZE parameter in the SNACOMM command in the profile command file, INPRO. Retry the program.

6200 ENABLESSL invalid on SSL command.

Explanation: You specified an invalid value for the ENABLESSL parameter in the SSL command. ENABLESSL must be a y or n.

User Response: Correct the ENABLESSL parameter. Retry the program.

6201 CIPHERSUITES invalid on SSL command.

Explanation: You specified an invalid value for the CIPHERSUITES parameter in the SSL command. The value of the CIPHERSUITES parameter is a string consisting of 1 or more 2-character values. Valid characters are **0-9** or **a-f**.

User Response: Correct the CIPHERSUITES parameter in the SSL command in the profile command file, INPRO. Retry the program. Valid values for CIPHERSUITES are determined when System SSL is installed on your MVS system and a default value is established. You should never change this value unless requested to do so by Help desk.

6202 SSLVERSION invalid on SSL command.

Explanation: You specified an invalid value for the SSLVERSION parameter in the SSL command. Valid values for SSLVERSION are **3031**, **30**, or **31**.

User Response: Correct the SSLVERSION parameter in the SSL command in the profile command file, INPRO. Retry the program. The default value is **3031**. You should never change this value unless requested to do so by Help desk.

6203 SSL only valid with TCP/IP communications.

Explanation: You specified Y for the ENABLESSL parameter, but the communication type is not TCP/IP.

User Response: Set your communication type to TCP/IP (COMMTYPE(T)) on the TRANSMIT command in the profile command file, INPRO, or set ENABLESSL to N, and retry the program.

Parser errors

This section describes the return codes for parser errors.

14000 Null character in input file.

Explanation: There is a null character in either the profile command file, INPRO, or the message command file, INMSG. Null characters are not permitted in INMSG or INPRO.

User Response: Check the message response file, OUTMSG, the profile response file, OUTPRO, or the response work file, OUTWORK, to determine which command produced the error. Correct the command file and retry the program.

14020 Command or parameter name too long.

Explanation: A command or parameter name in the command file is invalid because it is too long.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the command file and retry the program.

14030 Parameter value too long.

Explanation: A parameter value in the command file is invalid because it is longer than the maximum length permitted for the parameter. This is sometimes caused by unbalanced parentheses.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the appropriate command file and retry the program.

14040 Command expected but not found.

Explanation: You did not specify an expected command in the command file. It is possible that you specified a parameter before a command.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the appropriate command file and retry the program.

14050 Parameter and value or semicolon expected but not found.

Explanation: There is a syntax error in the profile command file, INPRO, or message command file, INMSG. Either a semicolon or a parameter and value was expected but was not found. This is usually caused by omitting the semicolon from a command, omitting the value of a parameter, or leaving spaces between the parameter name and value.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the appropriate command file and retry the program.

15010 Parameter value too long.

Explanation: The parameter value for one of the commands in the command file is longer than the maximum allowed for that parameter.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the appropriate command file and retry the program.

15020 Duplicate parameter found.

Explanation: The same parameter was specified more than once in a command.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the appropriate command file and retry the program.

15030 Invalid parameter found.

Explanation: You specified an invalid parameter in the command file.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the appropriate command file and retry the program.

15040 Command not recognized.

Explanation: You specified an unrecognized command in the command file.

User Response: Check the message response file, OUTMSG, profile response file, OUTPRO, or response work file, OUTWORK, to determine which command produced the error. Correct the appropriate command file and retry the program.

15042 COMMIT command only valid with user-initiated recovery.

Explanation: You specified a COMMIT command with the RECOVERY parameter on the TRANSMIT command specified as C, S, or F. COMMIT commands are processed with 'U' for RECOVERY only.

User Response: Update the RECOVERY parameter on the TRANSMIT command to 'U,' or remove all COMMIT commands from the message command file, INMSG. Retry the program.

CAUTION: If you reset the session using the RESET parameter on the IEBASE exec statement, you will no longer be able to continue the session. If it is necessary to reset the session, modify the previous message command file, INMSG, deleting any commands that have already been processed. Failure to do this may result in some data being lost or duplicated.

Destination verification errors

This section describes the return codes for errors that occur verifying the Information Exchange destination.

16020 The destination specified is not a valid IE destination.

Explanation: The destination specified in the SEND, SENDSTREAM, or SENDEDI command does not exist. The data was not sent.

User Response: Correct the destination in the SEND or SENDSTREAM command, EDI data, EDI destination table, or EDI qualifier table. Retry the command.

16030 IE destination is blocked by trading partner list or payment levels.

Explanation: The destination in the SEND, SENDSTREAM, or SENDEDI command exists. However, the message cannot be sent because it is blocked by the payment level specified or by the trading partner list. The data was not sent.

User Response: Make sure the Information Exchange destination and payment levels are correct. Check with your service administrator to change a trading partner list or payment levels. Retry the command.

16040 IE was unable to verify the destination immediately.

Explanation: The destination in the SEND, SENDSTREAM, or SENDEDI command could not be verified immediately because it is on another Information Exchange system. The data was not sent.

User Response: Retry the command using either an f, g, or n value for the VERIFY parameter. Information Exchange cannot immediately verify a destination on another system.

16050 No update access to library.

Explanation: You specified 'Y' for VERIFY on the PUTMEMBER command and either the library does not exist or you do not have update access to the library. The data was not sent.

User Response: Use Information Exchange Administration Services to verify that the library you are trying to update exists and that you have update authority for it. Retry the command.

16052 File with specified message key does not exist.

Explanation: The message key you specified does not match any of the files in your mailbox.

User Response: Verify that you specified the correct message key. Valid message keys are shown on the AVAILABLE record in response to a QUERY command. Correct the message key and retry the command.

16054 File to purge is being received.

Explanation: The message key you specified is for a file that is in the process of being received, and cannot be purged.

User Response: If you do not want to receive the file, then discontinue the receive process and reset the Information Exchange session. Retry the command.

16056 Information Exchange profile does not allow files to be purged.

Explanation: Your Information Exchange profile does not allow purging of files from your mailbox.

User Response: If you do not want to be able to purge files from your mailbox, use Information Exchange Administration Services or ask your System Administrator to change your profile to allow this action. Retry the command.

16060 Unable to retrieve library member.

Explanation: Information Exchange could not retrieve the library member because either the library does not exist, you do not have read access to the library, or the destination you specified is invalid.

User Response: Receive the system error message from your mailbox. The system error message explains why the GETMEMBER failed. Correct the problem and retry the command.

EDI errors

This section describes the return codes for EDI errors.

17102 Invalid X12 header in file %.54s.

Explanation: The X12 header in the data you attempted to send is invalid. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the X12 ISA header that caused the error in the envelope, and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17106 Missing X12 destination in file %.54s.

Explanation: The X12 header in the data you attempted to send does not contain an X12 receiver ID. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the X12 ISA header that caused the error in the envelope, and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17108 Invalid X12 destination in file %.54s.

Explanation: The X12 header in the data you attempted to send contains an X12 receiver ID that could not be resolved. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDI qualifier table, the EDI destination table, or the X12 ISA header that caused the error in the envelope. Retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17110 Invalid X12 binary or encrypted segment in file %.54s.

Explanation: The X12 data contains an invalid binary or security segment. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the X12 data and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17112 Invalid X12 binary or encrypted length in file %.54s.

Explanation: The length element in an X12 binary or security segment is invalid. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the X12 data and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17114 X12 control number is missing or invalid.

Explanation: The X12 header in the data you attempted to send does not contain an interchange control number, or the interchange control number is not numeric. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the X12 ISA header that caused the error in the envelope and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17116 X12 control number error.

Explanation: The X12 control number in the IEA is missing or does not match the control number in the ISA. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the X12 data that caused the error and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17118 Missing IEA in X12 data.

Explanation: The X12 IEA segment was not found in the data. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the X12 data that caused the error and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17202 Invalid UCS header in file %.54s.

Explanation: The UCS BG header in the data you attempted to send is invalid. The segment terminator must be hex 15. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the UCS BG header that caused the error and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17206 Missing UCS destination in file %.54s.

Explanation: The UCS header in the data you attempted to send does not contain a UCS receiver ID. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the UCS BG header that caused the error and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17208 Invalid UCS destination in file %.54s.

Explanation: The UCS BG header in the data you attempted to send contains a UCS receiver ID that could not be resolved. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDI qualifier table, the EDI destination table, or the UCS BG header that caused the error, and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17302 Invalid EDIFACT header in file %.54s.

Explanation: The EDIFACT UNB header in the data you attempted to send is invalid. The segment terminator was not found. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDIFACT UNB header in the envelope that caused the error and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17306 Missing EDIFACT destination in file %-.54s.

Explanation: The EDIFACT UNB header in the data you attempted to send does not contain an EDIFACT receiver ID. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDIFACT UNB header that caused the error, and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17308 Invalid EDIFACT destination in file %.54s.

Explanation: The EDIFACT UNB header in the data you attempted to send contains an EDIFACT receiver ID that could not be resolved. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDI qualifier table, the EDI destination table, or the EDIFACT UNB header in the envelope that caused the error. Retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17316 EDIFACT control number error.

Explanation: The EDIFACT control number in the UNZ is missing or does not match the control number in the UNB. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDIFACT data in the envelope that caused the error and retry the command. If there were multiple envelopes in this file, check the response file for SENT records to determine which envelopes in the file were sent.

17318 Missing UNZ in EDIFACT data.

Explanation: A UNA or UNB was found before the preceding EDIFACT envelope ended. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDIFACT data that caused the error and retry the command. If there were multiple envelopes in this file, check the response file for SENT records to determine which envelopes in the file were sent.

17402 Invalid UN/TDI header in file %.54s.

Explanation: The UN/TDI STX header in the data you attempted to send is invalid. The segment terminator was not found. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the UN/TDI STX header that caused the error, and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17406 Missing UN/TDI destination in file %.54s.

Explanation: The UN/TDI STX header in the data you attempted to send does not contain a UN/TDI receiver ID. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the UN/TDI STX header that caused the error, and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

17408 Invalid UN/TDI destination in file %-.54s.

Explanation: The UN/TDI STX header in the data you attempted to send contains a UN/TDI receiver ID that could not be resolved. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDI qualifier table, the EDI destination table, or the UN/TDI STX header in the envelope that caused the error. Retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

18010 Null character found in EDI table "%54s".

Explanation: You specified a null character in the EDI destination table or the EDI qualifier table specified.

User Response: Correct the table and retry the command.

18020 EDI table %.54s has an invalid parameter "%.9s".

Explanation: The parameter name in the EDI destination table or EDI qualifier table is not a valid parameter name or does not have an associated value.

User Response: Correct the table and retry the command.

18030 EDI table %.54s has an invalid parameter value "%.9s".

Explanation: The parameter value in the EDI destination table or EDI qualifier table is longer than the maximum length allowed.

User Response: Correct the table and retry the command.

18040 EDI table %.54s contains a duplicate parameter "%.9s".

Explanation: You specified the same parameter more than once for an entry in the EDI destination table or EDI qualifier table. This may be caused by a missing semicolon between entries.

User Response: Correct the table and retry the program.

18110 End of file found before end of EDI envelope in file "%.54s".

Explanation: Expedite Base encountered the end of the file before the end of the EDI envelope. This envelope was not sent.

User Response: Correct the EDI data and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

18120 Unable to determine EDI type in file %.54s.

Explanation: Expedite Base could not determine the type of EDI data you attempted to send. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDI data and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

18130 Error processing EDI data in file %-.54s.

Explanation: There was an error processing the EDI data you attempted to send. This envelope and the envelopes following it in the file were not sent.

User Response: Correct the EDI data and retry the command. If there were multiple envelopes in this file, check the message response file, OUTMSG, for SENT records to determine which envelopes in the file were sent.

18210 Qualifier table %.54s contains an invalid alias "%5.4s".

Explanation: You specified an invalid value for the default alias in the qualifier table. The first character of the ALIAS is the destination table type and must be blank, g, o or p. The last three characters of the ALIAS are the destination table ID and must be one to three alphanumeric characters.

User Response: Correct the EDI destination table and retry the program.

18230 Invalid IE destination for entry "%.35s".

Explanation: You specified an invalid Information Exchange destination for the EDI destination in the EDI destination table. This indicates that the destination is missing, incomplete, contains more than one destination type, or uses an invalid alias type.

User Response: Correct the EDI qualifier table and retry the program.

18250 End of file found before end of destination in file %.54s.

Explanation: Expedite Base encountered an end-of-file in the middle of either a qualifier table entry or EDI destination table entry. Make sure the last table entry ends with a semi-colon.

User Response: Correct the EDI qualifier table and retry the program.

18300 Invalid EDI data received.

Explanation: The data received with the RECEIVEEDI command was not valid EDI data. Once Expedite Base determines that the data was not valid EDI data, the rest of the data in the message is not reformatted. Therefore, the records may not be separated as you want them.

User Response: No response is needed. This is only an informational message.

TCP/IP communication errors

This section describes the return codes for TCP/IP communication errors.

19006 Unable to create socket. %s.

Explanation: Expedite Base was unable to create a socket for communication with Information Exchange.

User Response: Verify that TCP/IP has been configured properly on your system. Retry the program.

19007 Unable to connect to Information Exchange.

Explanation: Expedite Base was unable to connect to Information Exchange.

User Response: Information Exchange or the Domain Name Server may be temporarily unavailable. Verify that IETCPHOST*n* and IETCPPORT*n* are valid. Retry the program.

19009 Unable to resolve host name.

Explanation: Expedite Base was unable to resolve the host name.

User Response: Expedite Base tried to resolve the host name specified in IETCPHOSTn but was not successful. The host may be temporarily unavailable. Retry the program. If the problem persists, contact Help Desk.

19011 Unable to connect to host. %s. %s.

Explanation: Expedite Base was unable to connect to the Information Exchange Common Front End. If you have never successfully connected to the Information Exchange Common Front End using TCP/IP, your connection may not be set up properly.

User Response: If you have connected before, verify that IETCPHOST*n* and IETCPPORT*n* are valid. The host may be temporarily unavailable. Retry the program. If the problem persists, contact Help Desk.

19012 TCP/IP subsystem is not running.%s.%s

Explanation: TCP/IP subsystem on your system is not running.

User Response: Verify that TCP/IP has been installed, configured and started properly. Retry the program. If the problem persists, contact Help desk.

19014 Invalid Information Exchange account and/or user ID specified.%s.%s

Explanation: You specified an account and user ID that is not valid.

User Response: Enter the correct account and user ID and run the session again.

19015 Invalid Information Exchange password specified.%s.%s

Explanation: You specified a password that is invalid.

User Response: Enter the correct password and run the session again.

19016 Unable to receive from host. %s. %s.

Explanation: Expedite Base was unable to receive data from Information Exchange.

User Response: Verify that the link has not been dropped, your Information Exchange ID is not being used by someone else, and the inactivity timeout has not been reached. Retry the program. If the problem persists, contact Help desk.

19017 Unable to send to host. %s. %s.

Explanation: Expedite Base was unable to send data to Information Exchange.

User Response: Verify that the link has not been dropped, your Information Exchange ID is not being used by someone else, and the inactivity timeout has not been reached. Retry the program. If the problem persists, contact Help desk.

19019 IETCPHOSTn and IETCPPORTn missing on TCPCOMM command.

Explanation: You must specify both a IETCPHOST*n* and IETCPPORT*n* parameter in the TCPCOMM command.

User Response: Verify that you have a valid TCPCOMM command with at least one IETCPHOST*n* and IETCPPORT*n* parameter specified in the profile command file, INPRO. If you do not know the correct values for these parameters, contact Help desk. Retry the program.

19020 IETCPHOSTn or IETCPPORTn missing on TCPCOMM command.

Explanation: You must specify both IETCPHOSTn and IETCPPORTn in the TCPCOMM command. Only one was specified.

User Response: Verify that you have a valid TCPCOMM command with at least one IETCPHOSTn and IETCPPORTn parameter specified in the profile command file, INPRO. If you do not know the correct values for these parameters, contact Help desk. Retry the program.

19021 Invalid IP address.

Explanation: You specified an invalid IP address for the IETCPHOSTn parameter in the TCPCOMM command. A valid IP address uses the format nnn.nnn.nnn.nnn, where each *nnn* is a number from 1 to 255.

User Response: Correct the IP address specified in the IETCPHOSTn parameter specified in the profile command file, INPRO. If you do not know the correct IP address for Information Exchange, contact Help desk. Retry the program.

19022 Invalid IP port number.

Explanation: You specified an invalid IP port number in the IETCPPORTn parameter in the TCPCOMM command. A valid IP port number must be a number from 1 to 65535.

User Response: Correct the IP port number specified in the IETCPPORTn parameter in the profile command file, INPRO. If you do not know the correct IP port number for Information Exchange, contact Help desk. Retry the program.

19031 Invalid data received.

Explanation: Expedite Base/MVS received invalid data during session initialization.

User Response: Check to make sure that you have used the correct IP address for a non-SSL connection and retry the program.

File validation errors

This section describes the return codes for file validation errors.

20022 Invalid record length for INPRO file.

Explanation: The record length for the profile command file, INPRO, must be less than or equal to 255.

User Response: Correct your JCL or profile command file and try the program again.

20030 Invalid record length for OUTPRO file.

Explanation: The record length for the profile response file, OUTPRO, must be less than or equal to 255, greater than or equal to the record length of the profile command file, INPRO, and greater than or equal to 72.

User Response: Correct your JCL or profile response file, OUTPRO, and try the program again.

20038 Invalid record format for OUTPRO file.

Explanation: The record format for the profile response file, OUTPRO, indicates that the file contains ISO/ANSI device control characters or machine code control characters, for example, FBA or FBM. This record format is not supported for the profile response file.

User Response: Correct your JCL or profile response file, OUTPRO, and try the program again.

20040 Invalid record length for OUTMSG file.

Explanation: The record length for the message response file, OUTMSG, must be less than or equal to 255 and greater than or equal to the record length of the message command file, INMSG.

User Response: Correct your JCL or message response file, OUTMSG, and try the program again.

20042 Invalid device type for OUTMSG file.

Explanation: For checkpoint-level recovery, the message response file, OUTMSG, must be located on DASD. SYSOUT and tape data sets are not permitted.

User Response: Correct your JCL or the OUTMSG file and try the program again.

20044 Invalid data set organization for OUTMSG file.

Explanation: If using checkpoint-level recovery, the message response file, OUTMSG, must be a sequential data set, not a PDS member.

User Response: Correct your JCL or the OUTMSG file and try the program again.

20046 Invalid DISP specified for OUTMSG file.

Explanation: When using checkpoint-level recovery, DISP=MOD is not permitted in the message response file, OUTMSG.

Explanation: Correct your JCL and try the program again.

20048 Invalid record format for OUTMSG.

Explanation: The record format for the message response file, OUTMSG, indicates that the file contains ISO/ANSI device control characters or machine code control characters, for example, FBA or FBM. This record format is not supported for the message response file.

User Response: Correct your JCL or message response file, OUTMSG, and try the program again.

20052 Invalid record length for INMSG.

Explanation: The record length for the message command file, INMSG, must be less than or equal to 255.

User Response: Correct your JCL or message command file and try the program again.

20060 Invalid device type for work file %.54s.

Explanation: The work files used for checkpoint-level recovery must be located on DASD. Sysin, sysout, or tape data sets are not permitted.

User Response: Correct your JCL or the indicated work file and try the program again.

20062 **Invalid data set organization for work file %.54s.**

Explanation: The work files used for checkpoint-level recovery must be sequential data sets. PDS members are not permitted.

User Response: Correct your JCL or the indicated work file and try the program again.

20064 **Invalid DISP specified for work file %.54s.**

Explanation: DISP=MOD is not permitted for the checkpoint-level recovery work files.

User Response: Correct your JCL and try the program again.

General environment errors

This section describes the return codes for general environment errors.

20365 **Unable to allocate storage location id %.40s.**

Explanation: Expedite Base was unable to get the memory it needed. The location ID indicates the location within the program where the memory allocation failed. This ID can be used by Help desk if the memory allocation failure is caused by an internal error.

User Response: Make sure there is adequate memory and try the program again. If the problem persists, call Help desk.

20410 **Profile not found.**

Explanation: Expedite Base could not find the profile information. You must have a profile command file, INPRO.

User Response: Create the profile command file, INPRO, containing all of the required information, and include an INPRO DD card in the JCL to indicate its location. Then try the program again.

20611 **Error opening input file.**

Explanation: Expedite Base could not open the message command file, INMSG. There might be a REASON parameter that gives more information about why the file could not be opened.

User Response: If there is a REASON parameter, check it for more information about the error. Check your JCL and the INMSG file for errors and try the program again.

20620 **Unable to restart at checkpoint due to error in INMSG.**

Explanation: Commands in the message command file, INMSG, that were processed before the last checkpoint have been changed. Expedite Base is unable to continue the current session at the latest checkpoint. Commands in INMSG that have already been processed, echoed to OUTMSG, should not be changed if you want to restart the session at the last checkpoint.

User Response: Reset the session using the RESET parameter of the IEBASE EXEC statement. Before starting the next session, review the message response file, OUTMSG, to see which commands were processed successfully. Remove these commands from the message command file, INMSG, so they are not processed again.

CAUTION: If you reset the session using the RESET parameter, you will no longer be able to continue the previous session. Failure to modify the message command file, INMSG, before resetting the session may result in some data being lost or duplicated.

21000 Expedite Base encountered an unexpected condition.

Explanation: Expedite Base encountered an unexpected condition during execution.

User Response: Retry the program. If the problem persists, contact Help desk. You will be asked to fax or send the trace file BASETRC from the failed session to Help desk for problem determination.

21013 Error reading EDI table "%-.54s".

Explanation: Expedite Base could not read the EDI qualifier table or EDI destination table file.

User Response: Make sure there are no input or output problems with the file. Retry the program.

21606 Receiver not found in look up table.

Explanation: You specified 'T' for the COMPRESS parameter, which indicates you want the look up table to be consulted before your data is sent compressed, but the receiver was not in the look up table; therefore, the data was not compressed before sending. The COM-PRESS parameter in your OUTMSG file will be set to 'W' for warning.

User Response: Check the message response file, OUTMSG, or response work file, OUTWORK, to determine which receiver was not found in the table. Either specify 'Y' for COMPRESS or add the receiver to your look up table, and retry the program.

21810 File operation %s failed on file %.54s.

Explanation: Expedite Base could not access a file in the requested mode.

User Response: Check your file and make sure there are no I/O problems. Check the file permissions to make sure Expedite Base has access to the file. Retry the program. If the problem persists, contact Help desk. You will be asked to fax or send the trace file BASETRC to Help desk for problem determination.

22240 Error looking up message.

Explanation: An error occurred while looking up an error message. This is caused by an error in the error message description file, ERRORMSG, or the extended error text file, ERRORTXT.

User Response: If you modified either the error message description file, ERRORMSG, or the extended error text file, ERRORTXT, retry the program with the original files. If the problem continues, contact Help desk.

22411 Error opening receive file.

Explanation: Expedite Base could not open the file for the data being received. There might be a REASON parameter that gives more information about why the file could not be opened.

User Response: If there is a REASON parameter, check it for more information about the error. Make sure that you specified a valid FILEID for the RECEIVE or RECEIVEEDI command. Also, check the JCL, the receive file, and the RCVWORK file for errors. Try the program again.

22413 Invalid record length in the receive work file, RCVWORK.

Explanation: The record length of the RCVWORK file is not the same as it was in the session that is being restarted. The work files should not be modified while a session is in progress.

User Response: Make sure the RCVWORK file refers to the same data set that it did in the session that is being restarted and that the file has not been modified. If the problem continues, reset the session using the RESET parameter in the IEBASE EXEC statement.

CAUTION: If you reset the session using the RESET parameter, you can no longer continue the previous session, and some data might be lost.

22420 Record length mismatch in receive file.

Explanation: The records in the data received, as indicated by CRLF characters or 2-byte length indicators, were too long for the receiving file, and you specified 'E' for RESRECL in the RECEIVE command.

User Response: Receive the data into a file with a larger record length. If you want Expedite Base to split the records that are too long, specify 'S' for RESRECL in the RECEIVE command.

22440 Length delimiters in data invalid.

Explanation: The common data header or DELIMITED parameter indicated that the data contained 2-byte length delimiters to separate records, but the lengths indicated by the delimiters did not match the length of the data. The length delimiters were processed, but the records may not be separated as you want them.

User Response: No response is needed. This is only an informational message.

22460 Some of the records in the received file were segmented.

Explanation: Some of the records delimited by 2-byte length indicators or by CRLF characters in the received file were segmented into multiple records because they were too long.

User Response: No response is necessary. This is an information-only message.

22484 Invalid record length on receive.

Explanation: The record length of the receive file is not the same as it was in the session that is being restarted. The file for the receive in progress should not be modified while a session is in progress.

User Response: Make sure that the file ID used when receiving refers to the same data set that it did in the session being restarted and that the file has not been modified. If the problem continues, reset the session using the RESET parameter in the IEBASE EXEC statement.

CAUTION: If you reset the session using the RESET parameter, you can no longer continue the previous session, and some data might be lost.

22610 Send or Putmember file not found.

Explanation: Expedite Base could not open the file indicated in the SEND or PUTMEMBER command.

User Response: Check the FILEID parameter on the SEND or PUTMEMBER command and retry the command.

22615 Send or Putmember file was empty.

Explanation: The file indicated in the SEND or PUTMEMBER command was empty.

User Response: Correct the message command file, INMSG, the SEND file, or the PUTMEMBER file; then retry the command.

23410 EDI send file %.54s not found.

Explanation: Expedite Base could not open the file indicated in the SENDEDI command. There might be a REASON parameter that gives more information about why the file could not be opened.

User Response: If there is a REASON parameter, check it for more information about the error. Also, make sure the file specified is not empty. Check the FILEID parameter and JCL for errors and try the command again.

23415 EDI send file was empty.

Explanation: The file indicated in the SENDEDI command did not contain any EDI data. It was empty or contained only blanks.

User Response: Correct the message command file, INMSG, or EDI send file and retry the command.

23500 No libraries found to list.

Explanation: There were no libraries found for the parameters specified on the LISTLIBRARIES command.

User Response: Check the parameters specified in the LISTLIBRARIES command. If the AUTHORITY, SELECTION, or OWNER parameters are incorrect, correct them and retry the command.

23502 Owning account for libraries invalid.

Explanation: The owning account ID specified by the OWNER parameter of the LISTLIBRARIES command is not recognized by Information Exchange.

User Response: Check the owning account ID specified in the LISTLIBRARIES command is correct. If not, correct the OWNER parameter and retry the command.

23504 Library does not contain any members.

Explanation: The library specified in the LISTMEMBERS command does not contain any members. The command was not processed.

User Response: Check the library specified in the LISTMEMBERS command. Correct the LIBRARY parameter and try again.

23506 Library does not exist.

Explanation: The library specified in the LISTMEMBERS command does not exist on Information Exchange.

User Response: Check the library specified in the LISTMEMBERS command. If it is correct, use Information Exchange Administration Services to verify that the library exists.

23508 Library owning account invalid.

Explanation: The library owning account specified in the OWNER parameter of the LISTMEMBERS command is not recognized by Information Exchange.

User Response: Check the library owning account specified in the LISTMEMBERS command. Correct the OWNER parameter and retry the command.

23510 Read access not permitted for library.

Explanation: The ACCOUNT or USERID does not have read access to the library specified in the LISTMEMBERS command.

User Response: Use Information Exchange Administration Services to verify that you have read access to the library. Retry the command.

Session start and end errors

This section describes the return codes for session start and end errors.

24000 Error in restart processing.

Explanation: Expedite Base was not able to restart the session with Information Exchange. The session file, SESSION, may be damaged.

User Response: Reset the session using the RESET parameter on the IEBASE EXEC statement. Also, make sure there is not another user using this user ID. If the problem persists, contact Help desk. Before starting the next session, review the message response file, OUTMSG, to see which commands were processed successfully. Remove these commands from the message command file, INMSG, so they are not processed again.

CAUTION: If you reset the session using the RESET parameter, you will no longer be able to continue the previous session. Failure to modify the message command file, INMSG, before resetting the session may result in some data being lost or duplicated.

24020 Restart and original recovery levels are not equal.

Explanation: The restart level differs from the original recovery level. The session has not started. Either your session file, SESSION, is damaged or another user is using this user ID.

User Response: Reset the session using the RESET parameter on the IEBASE EXEC statement. Also, make sure there is not another user using this user ID. If the problem persists, contact Help desk. Before starting the next session, review the message response file, OUTMSG, to see which commands were processed successfully. Remove these commands from the message command file, INMSG, so they are not processed again.

CAUTION: If you reset the session using the RESET parameter, you will no longer be able to continue the previous session. Failure to modify the message command file, INMSG, before resetting the session may result in some data being lost or duplicated.

24100 Session and Information Exchange checkpoints do not match.

Explanation: In a session using checkpoint-level recovery, the checkpoint numbers for the send or receive side of the session do not match the values Information Exchange recorded. Your session file, SESSION, may be damaged.

User Response: Reset the session using the RESET parameter on the IEBASE EXEC statement. Also, make sure there is not another user using this user ID. If the problem persists, contact Help desk. Before starting the next session, review the message response file, OUTMSG, to see which commands were processed successfully. Remove these commands from the message command file, INMSG, so they are not processed again.

CAUTION: If you reset the session using the RESET parameter, you will no longer be able to continue the previous session. Failure to modify the message command file, INMSG, before resetting the session may result in some data being lost or duplicated.

24200 Invalid time zone.

Explanation: You specified an invalid time zone.

User Response: Correct the TIMEZONE parameter in the IDENTIFY command in the profile command file, INPRO, and retry the program. If the problem persists, contact Help desk.

24210 Invalid maximum segments.

Explanation: Expedite Base used an invalid value for the MAXMSGS parameter in the TRANSMIT command in the profile command file, INPRO. The valid range of values is 1 to 10.

User Response: Correct the MAXMSGS parameter in the TRANSMIT command in the profile command file and try the program again.

24220 Invalid reset field.

Explanation: Expedite Base used an invalid value for a field when trying to start the session with Information Exchange.

User Response: Contact Help desk.

24230 Invalid field on session start.

Explanation: Expedite Base used an invalid value for a field when trying to start the session with Information Exchange.

User Response: Ensure that your Information Exchange User Profile indicates that your user ID has the ability to reset sessions. If it does, contact Help desk for further assistance.

24270 Incorrect Information Exchange password.

Explanation: Your Information Exchange password is incorrect.

User Response: Correct the IEPASSWORD in the IDENTIFY command in the profile command file, INPRO, or in the START command in the message command file, INMSG, and retry the program. If you just changed your password, make sure you update the IDENTIFY or START command to reflect the new password.

24280 Invalid Information Exchange user ID.

Explanation: You specified an invalid user ID. Information Exchange does not recognize the account ID or user ID specified in the START command or the profile.

User Response: If a START command was used with ACCOUNT and USERID parameters, make sure they are correct. If the IEACCOUNT and IEUSERID are taken from the profile, make sure you specified them correctly in the IDENTIFY command in the profile command file, INPRO. If the problem continues, contact Help desk.

24290 Invalid new Information Exchange password.

Explanation: You are an Extended Security Option user, and you specified an invalid new Information Exchange password.

User Response: Correct the NIEPASSWORD parameter in the IDENTIFY command in the profile command file, INPRO, or in the START command in the message command file, INMSG, and retry the program. ESO passwords have special requirements. Refer to the product documentation for information about these requirements.

24300 Invalid password. User ID was revoked.

Explanation: You are an ESO user and sent three successive session starts to Information Exchange with incorrect passwords. The Information Exchange user ID has been revoked.

User Response: Contact your service administrator to request that your password be reset using Information Exchange Administration Services. Resetting the password restores the user ID.

24310 New password is required.

Explanation: You are an ESO user and did not specify a new password. If the Information Exchange password for an ESO user is the same as the Information Exchange user ID, the ESO user must specify a new password.

User Response: Use the NIEPASSWORD parameter of the IDENTIFY command in the profile command file, INPRO, or the START command in the message command file, INMSG, to change the password.

24320 Error starting session with Information Exchange.

Explanation: There was an error trying to start the session with Information Exchange.

User Response: Contact Help desk.

24600 Information Exchange did not return a valid session end response.

Explanation: The Information Exchange session may not have ended.

User Response: Try to restart the session. If the problem persists, contact Help desk.

24610 Information Exchange did not end the session due to an error.

Explanation: Information Exchange indicated that there was an error, and the session could not end properly.

User Response: Try to restart the session. If the problem persists, contact Help desk.

24620 Dequeue error.

Explanation: There was an error trying to dequeue an Information Exchange account ID or user ID.

User Response: If this occurred at the end of the command file, no action is needed. If you have multiple START and END commands in the command file, and it occurred on an earlier END command, try the program again. If the problem continues, contact Help desk.

Session errors

This section describes the return codes for session errors.

28000 Warnings generated for the command.

Explanation: This return code indicates that warning messages were generated during the command, but the command was able to complete.

User Response: Check the WARNING records in the message response file, OUTMSG, for details.

28010 The IE session completed normally but not all requests were processed.

Explanation: The Information Exchange session completed normally, but not all of the requests in the message command file, INMSG, processed, or some requests generated warnings.

User Response: Check the message response file, OUTMSG, to see which requests did not process normally. Correct those requests in a new INMSG file and retry the program.

28100 QUERY response indicates warning.

Explanation: Information Exchange found one or more errors in the QUERY command but was still able to process the command. Expedite Base may not write AVAILABLE records for some or all of the messages in your mailbox.

User Response: Check the error messages in your mailbox to see what caused the error. If needed, correct the error and try the command.

28120 GETMEMBER response indicates a warning.

Explanation: The response from Information Exchange to the GETMEMBER command shows that there was a warning while processing the command.

User Response: Make sure the library and member that you are trying to retrieve exist, and that you have access to them. If there is a system error message in your mailbox, retrieve it or view it via Information Exchange Administration Services to determine what caused the error. Correct the problem and try the command again.

28140 AUDIT response indicates warning.

Explanation: Information Exchange found one or more errors in the AUDIT command but was still able to process the command. However, the audit records in your mailbox might be different than requested.

User Response: Check the error messages in your mailbox to see what caused the error. If the audit file in the mailbox does not meet your requirements, correct the AUDIT command in the message command file, INMSG, and retry the program.

28141 AUDIT response indicates error.

Explanation: Information Exchange found one or more errors in the AUDIT command and was unable to process the command.

User Response: Check the error messages in your mailbox to see what caused the error. Correct the error and retry the command. No audit file has been placed in your mailbox.

28160 IE could not verify the destination.

Explanation: Information Exchange encountered an error when verifying the destination for the SEND, SENDSTREAM, or SENDEDI command. The data was not sent.

User Response: Try the command again. If the problem persists, contact Help desk.

28170 LISTLIBRARIES response indicates a warning.

Explanation: Information Exchange found one or more errors in the LISTLIBRARIES command but was still able to process the command. Expedite Base/MVS may not write LIBRARYLIST records for some or all of the libraries you have access to.

User Response: Check the error messages in your mailbox to see what caused the error. If needed, correct the error and retry the command.

28171 There are more files to be received.

Explanation: The session ended successfully, but there are more files in the mailbox to be received.

User Response: Process the data already received and run Expedite Base again to receive the additional data. If you switch to checkpoint-, user-, or file-level recovery, you will be able to receive all of the files in your mailbox in a single session without encountering the 28171 return code. For more information, see Chapter 6, “Using session-level recovery.”

28175 LISTMEMBERS response indicates a warning.

Explanation: Information Exchange found one or more errors in the LISTMEMBERS command but was still able to process the command. Expedite Base may not write MEMBERLIST records for some or all of the libraries you have access to.

User Response: Check the error messages in your mailbox to see what caused the error. If needed, correct the error and retry the command.

28180 PURGE response indicates an error.

Explanation: An unexpected error occurred while trying to process the PURGE command.

User Response: Retry the program. If the problem persists, contact Help desk.

28190 Invalid common data header received.

Explanation: Expedite Base received an invalid Common Data Header. The data was received and processed as if no CDH was received.

User Response: Verify that the file was received as expected. You may wish to inform the sender that the interface sent an invalid CDH. If the sending interface was an Expedite Base product, contact Help desk.

28200 COMMIT command only valid after a SEND, SENDEDI, or PUTMEMBER command.

Explanation: COMMIT command only valid after a SEND, SENDEDI or PUTMEMBER has been specified.

User Response: A COMMIT command was specified but there was no SEND, SENDEDI, or PUTMEMBER command preceding it. The COMMIT command did not initiate a commit.

APPC communications support errors

This section describes APPC communications support errors.

29800 Unable to allocate a conversation with the IE Common Front End.

Explanation: Expedite Base/MVS was unable to allocate an LU 6.2 conversation with the Information Exchange Common Front End. This error commonly occurs when the Information Exchange Common Front End is not available.

User Response: Try the Expedite Base/MVS program again. If the problem persists, make sure that your SNACOMM parameters are correct and have your VTAM system programmer make sure that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined and your SNACOMM parameters are correct, contact Help desk.

29801 Unable to deallocate a conversation with the IE Common Front End.

Explanation: Expedite Base/MVS was unable to deallocate an LU 6.2 conversation with the Information Exchange Common Front End.

User Response: Try the Expedite Base/MVS program again. If the problem persists, have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29802 Unable to establish a session with the IE Common Front End.

Explanation: Expedite Base/MVS was unable to establish a communication session with the Information Exchange Common Front End.

User Response: Try the Expedite Base/MVS program again later. If the problem persists, make sure that your SNACOMM parameters are correct and have your VTAM system programmer make sure that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined and your SNACOMM parameters are correct, contact Help desk.

29803 APPC display command failure.

Explanation: An APPC display command failed. This problem can occur if you are using an LU name that was defined for ExpEDIte/MVS Host Release 3.

User Response: Try the Expedite Base/MVS program again. If the problem persists, have your VTAM system programmer verify that your VTAM application is correctly defined and activated. Make sure that the names you specified in the IELUNAME and USERLUNAME parameters of the SNACOMM command were not originally defined for expEDIte/MVS Host Release 3. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29804 APPC RECEIVE command failure.

Explanation: An APPC RECEIVE command failed. There might have been a communication link failure.

User Response: Try the Expedite Base/MVS program again. If the problem persists, have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29805 APPC SEND command failure.

Explanation: An APPC SEND command failed. There might have been a communication link failure.

User Response: Try the Expedite Base/MVS program again. If the problem persists, have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29806 Unable to shut down Expedite Base/MVS's sessions.

Explanation: The command to set Expedite Base/MVS's sessions to zero as part of normal shutdown has failed.

User Response: Try the Expedite Base/MVS program again. If the problem persists, have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29807 Unauthorized conversation initiated.

Explanation: An unknown program attempted to start a conversation with Expedite Base/MVS.

User Response: Try the Expedite Base/MVS program again later. If the problem persists, make sure there are no other programs on your system attempting to communicate with the LU name defined in the USERLUNAME parameter of your SNACOMM command. Also have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29808 Unauthorized bind detected.

Explanation: An unknown application attempted to bind a session with Expedite Base/MVS.

User Response: Try the Expedite Base/MVS program again. If the problem persists, make sure there are no other programs on your system attempting to communicate with the LU name defined in the USERLUNAME parameter of your SNACOMM command. Also have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29809 Unauthorized conversation initiated.

Explanation: An unknown program attempted to start a conversation with Expedite Base/ MVS.

User Response: Try the Expedite Base/MVS program again later. If the problem persists, make sure there are no other programs on your system attempting to communicate with the LU name defined in the USERLUNAME parameter of your SNACOMM command. Also, have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29810 Unable to establish a session with the IE Common Front End.

Explanation: Expedite Base/MVS was unable to establish a communication session with the Information Exchange Common Front End.

User Response: Try the Expedite Base/MVS program again. If the problem persists, take the following steps:

1. Make sure that the IELUNAME parameter of your SNACOMM command is spelled correctly.
2. Make sure that the IELUNAME parameter that you are using specifies the Information Exchange Common Front End. In the U.S., this is IBM0RELY. You cannot connect directly to Information Exchange or to the Information Exchange Common Front End.
3. Make sure that the LU name that you specified in the USERLUNAME parameter of the SNACOMM command has been defined to the Information Exchange Common Front End. This should have happened when you ordered Expedite Base/MVS.
4. Have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active.
5. If none of the above actions solves the problem, contact Help desk.

29811 Unknown IE Common Front End LU Name.

Explanation: The LU name you specified in the IELUNAME parameter of the SNACOMM command is unknown to VTAM.

User Response: Try the Expedite Base/MVS program again. If the problem persists, make sure that the IELUNAME parameter of your SNACOMM command is correct. Have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your SNACOMM parameters and your VTAM application are correctly defined, contact Help desk.

29812 Unknown mode name.

Explanation: The name you specified in the IELUMODE parameter of the SNACOMM command is unknown to VTAM.

User Response: Try the Expedite Base/MVS program again. If the problem persists, make sure that the IELUMODE parameter of your SNACOMM command is correct. In the U.S., IELUMODE should be IINAPPC, which is the default. Have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your SNACOMM parameters and your VTAM application are correctly defined, contact Help desk.

29813 Unknown mode name.

Explanation: The name you specified in the IELUMODE parameter of the SNACOMM command is unknown to VTAM.

User Response: Try the Expedite Base/MVS program again. If the problem persists, make sure that the IELUMODE parameter of your SNACOMM command is correct. In the U.S., IELUMODE should be IINAPPC, which is the default. Have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your SNACOMM parameters and your VTAM application are correctly defined, contact Help desk.

29814 Time out waiting for a session with the IE Common Front End.

Explanation: Expedite Base/MVS timed out waiting for an LU 6.2 session with the Information Exchange Common Front End. You need a session on the Information Exchange Common Front End for every concurrent user of Expedite Base/MVS. For example, if five people will be using Expedite Base/MVS at the same time, you need five sessions on the Information Exchange Common Front End. This error occurs when there are more concurrent users of Expedite Base on your system than there are available sessions on the Information Exchange Common Front End.

User Response: Retry the program when fewer people are running Expedite Base/MVS on your system. If you normally have this many people using Expedite Base at the same time, contact Help desk to increase the number of sessions available to you on the Information Exchange Common Front End.

29820 CSECT "ConvTask" missing from loadmodule.

Explanation: CSECT "ConvTask," which contains the programming necessary to send and receive information over an APPC conversation, is not present in the IEBASE load module. This indicates that the product has not been installed correctly.

User Response: Contact the system programmer responsible for the installation of Expedite Base/MVS for verification of correct installation.

29821 Conversation subtask abend.

Explanation: The APPC Conversation Subtask has abnormally ended.

User Response: This indicates an error has occurred in the Expedite Base/MVS product. A diagnostic message indicating the contents of the subtask's ECB is displayed on either the system log through a WTO macro, or in the Expedite Base/MVS trace file if it is active. Obtain an Expedite Base/MVS link trace and MVS dump of the failing task by running the job again with a SYSUDUMP DD card if needed before contacting Help desk.

29822 Receive Timeout.

Explanation: Expedite Base/MVS did not receive an expected response within the time limit that you specified on the TIMEOUT parameter of the TRANSMIT command. This can be caused by another user communicating with Information Exchange using your account and user ID.

User Response: Try the Expedite Base/MVS program again. If the problem persists, make sure that no other users are using your Information Exchange account and user ID. You might want to increase your TIMEOUT value, especially if you have a large number of files in your Information Exchange mailbox. If increasing your TIMEOUT parameter does not help, contact Help desk.

29826 CSECT “VTAMTask” missing from loadmodule.

Explanation: CSECT "VTAMTask," which contains the programming necessary to start and end the VTAM application, is not present in the IEBASE load module. This indicates that the product has not been installed correctly.

User Response: Contact the system programmer responsible for the installation of Expedite Base/MVS for verification of a correct installation.

29827 VTAM subtask abend.

Explanation: The APPC Conversation Subtask has abnormally ended.

User Response: This indicates an error has occurred in the Expedite Base/MVS product. A diagnostic message indicating the contents of the subtask's ECB is displayed on either the system log using a WTO macro, or in the Expedite Base/MVS trace file if it is active. Obtain an Expedite Base/MVS link trace and MVS dump of the failing task by running the job again with a SYSUDUMP DD card if needed before contacting Help desk.

29840 ATTACH macro failed.

Explanation: An ATTACH macro has failed. This probably indicates an error in Expedite Base/MVS or in your MVS operating system.

User Response: Run the job again and, if the macro fails, contact Help desk.

29841 ENQ macro failed.

Explanation: An ENQ macro has failed. This probably indicates an error in Expedite Base/MVS or in your MVS operating system.

User Response: Run the job again and, if the macro fails, contact Help desk.

29843 STIMERM macro failed.

Explanation: A STIMERM macro has failed. This probably indicates an error in Expedite Base/MVS or in your MVS operating system.

User Response: Run the job again and, if the macro fails, contact Help desk.

29844 MODECB macro failed.

Explanation: A MODECB macro has failed. This probably indicates an error in Expedite Base/MVS or in your MVS operating system.

User Response: Run the job again and, if the macro fails, contact Help desk.

29845 VTAM OPEN failed.

Explanation: The OPEN macro issued for your LU name has failed. This can occur if you have an invalid USERLUNAME on your SNACOMM command.

User Response: Try the Expedite Base/MVS program again. If the problem persists, make sure that you have specified the correct USERLUNAME in your SNACOMM command. Also, have your VTAM system programmer verify that your VTAM application is correctly defined and activated. A diagnostic message that might help your VTAM system programmer is displayed on either the system log using a WTO macro, or at the end of the Expedite Base/MVS trace file if it is active. If your VTAM application is correctly defined, contact Help desk.

29846 SETLOGIN macro failed.

Explanation: A SETLOGON macro has failed. This probably indicates an error in Expedite Base/MVS or in your MVS operating system.

User Response: Run the job again and, if the macro fails, contact Help desk.

29847 DEQ macro failed.

Explanation: A DEQ macro has failed. This probably indicates an error in Expedite Base/MVS or in your MVS operating system.

User Response: Run the job again and, if the macro fails, contact Help desk.

29862 Unable to locate conversation control block.

Explanation: Expedite Base/MVS was unable to locate an internal conversation control block. This probably indicates an error in Expedite Base/MVS.

User Response: Run the job again and, if the macro fails, contact Help desk.

29863 Unable to locate main control block.

Explanation: Expedite Base/MVS was unable to locate the main internal communication control block. This probably indicates an error in Expedite Base/MVS.

User Response: Run the job again and, if the macro fails, contact Help desk.

29864 Unable to locate transaction control block.

Explanation: Expedite Base/MVS was unable to locate its internal transaction control block. This probably indicates an error in Expedite Base/MVS.

User Response: Run the job again and, if the macro fails, contact Help desk.

29865 GDS prefix is unrecognizable.

Explanation: The GDS, generalized data stream, which contains the data received from Information Exchange, is not recognizable to Expedite Base/MVS.

User Response: If the Expedite Base/MVS trace file is active, the first 64 bytes of the data are recorded in the file. If the trace is not active, the data is written to the system log using a WTO macro. Have this data ready when contacting Help desk.

29866 Received data length is incorrect.

Explanation: The total length of the components of the GDS, generalized data stream, received from Information Exchange do not match the length of the data received from APPC. This indicates that an error has occurred in either Expedite Base/MVS or the Information Exchange Common Front End application.

User Response: If the Expedite Base/MVS trace file is active, the first 64 bytes of the data are recorded in the file. If the trace is not active, the data is written to the system log using a WTO macro. Have this data ready when contacting Help desk.

29999 Session end response failure.

Explanation: Expedite Base did not receive a session end response from Information Exchange, or a communication failure occurred upon receiving the session end response.

User Response: Follow these steps after a session that fails with 29999 to see if the previous session has completed.

1. Specify AUTOSTART(N), AUTOEND(N) and RECOVERY(S) on your TRANSMIT command in the Expedite Base profile.
2. Create an input file containing START and END records. An example follows:

   ```
   START CHECK(Y);
   END;
   ```

   Do not specify any other commands in the input file if you specify CHECK(Y) on the session start command.
3. Run Expedite.

   No data will be transferred in the above example, and you will not be charged for this inquiry.

4. Examine the output file to check the LASTSESS parameter value on the STARTED record.
    - LASTSESS(0) indicates that the previous session was successful. No further recovery is required.
    - LASTSESS(1) indicates that the previous session was not successful.

    If Expedite Base reported the 29999 Session End return code for a session, you should switch to checkpoint, file-level, or user-level recovery instead of session-level recovery for future sessions with a similar number of commands.

SSL communication errors

This section describes the return codes for SSL errors.

30006 Invalid common name on returned certificate.

Explanation: The certificate returned during the SSL handshake contains an invalid common name.

User Response: The common name that is returned has been written to the BASETRC file if you have requested a trace and the file exists. If the problem persists, turn on tracing, rerun the job, and contact Help desk for help in determining the cause of this error.

30007 Un-trusted certificate received.

Explanation: The certificate returned during the SSL handshake matches a certificate listed in the CERTFIL dataset.

User Response: The certificate serial number that was returned has been written to the BASETRC file, if you requested tracing and the file exists. The partner certificate serial number matches a serial number listed in the CERTFIL file. If the problem persists, turn on tracing, rerun the job, and contact Help desk for help in determining the cause of this error.

30008 User not allowed through the Secure Front End (SFE) Gateway.

Explanation: The user has no access privileges to the Secure Front End Gateway.

User Response: Check that you have specified the correct IP address for the Secure Front End Gateway that you are allowed to communicate with. If you believe that you have the correct IP address, contact Help desk instructions on how to proceed.

30009 Account/Userid does not match certificate's account/userid.

Explanation: The user information found in the X.509 certificate does not match the IE system.account.userid that was specified on the IDENTIFY or START command.

User Response: Make sure that the information you gave when you requested the certificate matches the IE system, account, and userid that you have requested on the IDENTIFY or START command. Correct the error and retry the program.

30010 Invalid Certificate.

Explanation: The certificate is not valid for use on this Secure Front End Gateway.

User Response: Make sure that the certificate specified was generated by the IGS PKI Profile Server. Correct the error and retry the program. If the problem persists, turn on tracing, rerun the program, and contact Help desk for help in determining the cause of this error.

30011 Unable to negotiate security specifications.

Explanation: The SSL negotiation did not complete with a satisfactory protection level for the SFE Gateway.

User Response: If the problem persists, turn on tracing, rerun the program, and contact Help desk for help in determining the cause of this error.

30012 Connection Temporarily refused.

Explanation: The connection to the SFE Gateway has been temporarily refused.

User Response: If the problem persists, turn on tracing, rerun the program, and contact Help desk for help in determining the cause of this error.

30013 SSL unknown error.

Explanation: An unexpected SSL error has occurred.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30014 Invalid key ring file.

Explanation: The file name specified in the KEYRINGFILE parameter of the IDENTIFY or START command was not found or could not be opened.

User Response: Verify that the filename is typed correctly and the permissions of the file are set so the file is readable.

30015 Invalid key ring stash file.

Explanation: The file name specified in the KEYRINGSTASHFILE parameter of the IDENTIFY or START command was not found or could not be opened.

User Response: Verify that the filename is typed correctly and the permissions of the file are set so the file is readable.

30100 SSL API Error: gsk_environment_open error

Explanation: The file name specified in the KEYRINGSTASHFILE parameter of the IDENTIFY or START command was not found or could not be opened.

User Response: Verify that the filename is typed correctly and the permissions of the file are set so the file is readable.

30102 SSL API Error: KEYRINGFILE error.

Explanation: The SSL API encountered an error with the KEYRINGFILE parameter.

User Response: Check that the KEYRINGFILE parameter is present and correct. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30103 SSL API Error: KEYRINGPASSWORD or KEYRINGSTASHFILE error.

Explanation: The SSL API reported a problem with the KEYRINGPASSWORD or KEYRINGSTASHFILE parameters.

User Response: Check that the KEYRINGPASSWORD or KEYRINGSTASHFILE parameter is present and correct. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30104 SSL API Error: certificate error.

Explanation: The SSL API reports a problem with the certificate.

User Response: Check the certificate. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30105 SSL API Error: CIPHERSUITES error.

Explanation: The SSL API encountered a parameter error with the CIPHERSUITES parameter.

User Response: Check the CIPHERSUITES parameter to make sure it is correct. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30106 SSL API Error: SSLVERSION error.

Explanation: The SSL API reports a problem with the SSLVERSION parameter.

User Response: Check the SSLVERSION parameter to make sure it is correct. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30107 SSL API Error: AUTHORIZATION error.

Explanation: The SSL API encountered a certificate authorization error.

User Response: Check the certificate to make sure it is correct. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30108 SSL API Error: session type error.

Explanation: The SSL API encountered a session type error.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30109 SSL API Error: gsk_environment_init error.

Explanation: The SSL API encountered an error on the gsk_environment_init function call.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30110 SSL API Error: gsk_secure_socket_init error.

Explanation: The SSL API encountered an error on the gsk_secure_soc_init function call.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30111 SSL API Error: gsk_secure_socket_read error.

Explanation: The SSL API encountered an error on the gsk_secure_socket_read function call.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30112 SSL API Error: gsk_secure_soc_write error.

Explanation: The SSL API encountered an error on the gsk_secure_soc_write function call.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30114 SSL API Error: gsk_secure_socket_open error.

Explanation: The SSL API encountered an error on the gsk_secure_socket_open function call.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30115 SSL API Error: socket descriptor error.

Explanation: The SSL API encountered a socket descriptor error.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30116 SSL API Error: keyring file label error.

Explanation: The SSL API encountered a keyring file label error.

User Response: If the problem persists, turn tracing on, retry the program, and contact Help desk.

30117 SSL API Error: Cache error.

Explanation: The SSL API encountered a cache error.

User Response: Check the cache size and make sure it is correct. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30118 SSL API Error: certificate authority not trusted.

Explanation: The certificate authority is not trusted.

User Response: Check the certificate authority you have and make sure it is correct. If the problem persists, turn tracing on, retry the program, and contact Help desk.

30119 SSL API Error: No permission to KDB file.

Explanation: You do not have permission to the KDB file.

User Response: Check to make sure your userid has permission to the KDB file and path, if necessary. If the problem persists, turn tracing on and retry the program, and contact Help desk.

Unexpected Errors

This section describes the return codes for unexpected errors.

31000 Unexpected condition found.

Explanation: Expedite Base has encountered an unexpected condition.

User Response: Contact Help desk. You will be asked to fax or send your BASETRC file to Help desk for problem determination.

31344 Invalid message inquiry response.

Explanation: Expedite Base received an unexpected error from Information Exchange while processing a MSGINFO command.

User Response: Try the program again later. If the problem persists, contact Help desk.

31346 Invalid session inquiry response.

Explanation: Expedite Base received an unexpected error from Information Exchange while processing a SESSIONINFO command.

User Response: Try the program again later. If the problem persists, contact Help desk.

31360 Error indication received from Information Exchange.

Explanation: Expedite Base received an unexpected error from Information Exchange.

User Response: Wait and retry the program later. Also, make sure there is not another user using this user ID. If the problem persists, contact Help desk.

31810 File operation %s failed on file %.54s.

Explanation: A checkpoint recovery file was damaged.

User Response: Check your file and make sure there are no I/O problems. Check the file permissions to make sure Expedite Base has access to the file. Reset the session and retry the program. If the problem persists, contact Help desk. You will be asked to fax or send the trace BASETRC to Help desk for problem determination.

CAUTION: If you reset the session using the RESET parameter on the IEBASE exec statement, you will no longer be able to continue the previous session. Failure to modify the message command file, INMSG, before resetting the session may result in some data being lost or duplicated.

32000 Expedite Base encountered an unexpected condition.

Explanation: Expedite encountered an unexpected condition and is unable to continue.

User Response: Reset the session and retry the program. If the problem persists, contact Help desk. You will be asked to fax or send the trace file BASETRC to Help desk for problem determination.

CAUTION: If you reset the session using the IEBASE parameter on the IEBASE exec statement, you will no longer be able to continue the previous session. Failure to modify the message command file, INMSG, before resetting the session may result in some data being lost or duplicated.

Expedite Base/MVS error message identifiers to the operator console

This section describes Expedite Base/MVS error message identifiers to the operator console.

EXX4000 ENQ macro failed.

Explanation: The result of an MVS ENQ macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4001 STIMER macro failed.

Explanation: The result of an MVS STIMER macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4002 RU size must be >= 256, 256 used.

Explanation: The value specified in the rusize() parameter of the SNACOMM profile command is less than 256. A value of 256 has been used.

User Response: Contact Help desk.

EXX4003 CSECT "VTAMTask" missing from load module.

Explanation: A necessary portion of the IEBASE module is missing.

User Response: Contact Help desk to obtain help in correcting the load module.

EXX4004 IDENTIFY macro failed EP(VTAMTASK).

Explanation: The result of an MVS IDENTIFY macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4005 ATTACH macro failed.

Explanation: The result of an MVS ATTACH macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4006 CESECT "ConvTask" missing from load module.

Explanation: A necessary portion of the IEBASE module is missing.

User Response: Contact Help desk to obtain help in correcting the load module.

EXX4007 IDENTIFY macro failed EP(CONVTASK).

Explanation: The result of an MVS IDENTIFY macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4008 VTAM subtask ended.

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It indicates that one of the subtasks used for APPC communication has terminated.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log.

EXX4009 MODCB macro failed.

Explanation: The result of a VTAM MODCB macro was non-zero. This usually indicates a failure within MVS and/or VTAM.

User Response: Contact Help desk.

EXX4010 SHOWCB macro failed.

Explanation: The result of a VTAM SHOWCB macro was non-zero. This usually indicates a failure within MVS and/or VTAM.

User Response: Contact Help desk.

EXX4011 SETLOGON macro.

Explanation: The result of a VTAM SETLOGON macro was non-zero. This usually indicates a failure within MVS and/or VTAM.

User Response: Contact Help desk.

EXX4012 LcTPcb() call failed.

Explanation: A call to the Expedite Base/MVS internal routine to locate a Transaction Program Control Block has failed. This indicates a failure within Expedite Base/MVS.

User Response: Contact Help desk.

EXX4013 APPCCMD(OPRCNTL/DISPLAY).

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It reflects the results of a VTAM APPCCMD macro with CONTROL=OPRCNTL and QUALIFY=DISPLAY.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem. See the appropriate VTAM documentation for explanations of the return codes.

EXX4014 APPCCMD(OPRCNTL/CNOS).

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It reflects the results of a VTAM APPCCMD macro with CONTROL=OPRCNTL and QUALIFY=CNOS.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem. See the appropriate VTAM documentation for explanations of the return codes.

EXX4015 APPCCMD(ALLOC/ALLOCD).

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It reflects the results of a VTAM APPCCMD macro with CONTROL=ALLOC and QUALIFY=ALLOWED.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem. See the appropriate VTAM documentation for explanations of the return codes.

EXX4016 Conversation subtask ended.

Explanation: This is an informative message usually found only in the APPC trace. It indicates that one of the subtasks used for APPC communication has terminated.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log.

EXX4017 DEQ macro failed.

Explanation: The result of an MVS DEQ macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4018 LcConvCb() call failed.

Explanation: A call to the Expedite Base/MVS internal routine to locate a Conversation Control Block has failed. This indicates a failure within Expedite Base/MVS.

User Response: Contact Help desk.

EXX4019 STIMERM SET macro.

Explanation: The result of an MVS STIMERM SET macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4020 Timeout waiting for Conv_subtask.

Explanation: The subtask used to process APPC requests has not completed within the expected time.

User Response: If the error continues to happen, contact Help desk.

EXX4021 STIMERM CANCEL macro.

Explanation: The result of an MVS STIMERM CANCEL macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4022 Error data (sendd).

Explanation: An APPC Send Data request has received an FMH-7 from the partner LU (the Information Exchange host system). This indicates the partner LU has detected or encountered an error.

User Response: If the error recurs, contact Help desk.

EXX4023 Invalid error data received (sendd).

Explanation: The error data received in an FMH-7 is not formatted correctly.

User Response: Contact Help desk for help in resolving this problem.

EXX4024 Last data received was (sendd).

Explanation: When an error occurs in the APPC communication portion of Expedite Base/MVS, for diagnostic purposes, the last block of data that was sent to Information Exchange is displayed in this message.

EXX4025 GDS prefix is unrecognizable.

Explanation: The GDS portion of the error data received in an FMH-7 is not formatted correctly.

User Response: Contact Help desk for help in resolving this problem.

EXX4026 Received data length is incorrect.

Explanation: The data length embedded in a mapped conversation does not match the length of the actual data received. This will cause Expedite Base/MVS to terminate with an error.

User Response: Attempt to restart the job. If it continues to fail, contact Help desk.

EXX4027 Return code from Recvdata().

Explanation: The return code from the Expedite Base/MVS internal routine RECVDATA() is non-zero. If an error occurs, this message will be written to the job log for diagnostic purposes.

User Response: Contact Help desk.

EXX4028 Invalid FMH7 received.

Explanation: An FMH-7 received from the partner LU (Information Exchange) is not formatted correctly.

User Response: Contact Help desk for help in resolving this problem.

EXX4029 STIMERM macro failed.

Explanation: The result of an MVS STIMERM macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4030 Timeout waiting for VTAM task to end.

Explanation: A subtask used for APPC communication has not terminated within the expected time interval. This may be a result of the load on your host system or the communication network.

User Response: If the message occurs often, contact Help desk.

EXX4031 DETACH macro failed.

Explanation: The result of an MVS DETACH macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk.

EXX4032 appccmd(oprcntl/actsess).

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It reflects the result of a VTAM APOPCCMD macro with CONTROL=OPRCNTL and QUALIFY=ACTSESS.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem. See the appropriate VTAM documentation for explanations of the return codes.

EXX4033 APPCCMD(OPRCNTL/DACTSESS).

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It reflects the results of a VTAM APPCCMD macro with CONTROL=OPRCNTL and QUALIFY=DACTSESS.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem. See the appropriate VTAM documentation for explanations of the return codes.

EXX4034 Exit entered with RPLCNTDC RPLDNTSC.

Explanation: This is a diagnostic message written when Expedite Base/MVS's VTAM "SCIP" exit is entered. When configured properly, this should never happen.

User Response: Contact Help desk.

EXX4035 Received FMH5 from LU with mode.

Explanation: This is a diagnostic message written when Expedite Base/MVS's VTAM "ATTN" exit is entered with an FMH-5. When configured properly, this should never happen. The FMH-5 is rejected and processing continues.

User Response: Determine who is attempting to allocate a conversation (which creates the FMH-5) with Expedite Base/MVS.

EXX4036 APPCCMD(CONTROL/RCVFMH5).

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It reflects the results of a VTAM APPCCMD macro with CONTROL=CONTROL and QUALIFY=RCVFMH5.

User Response: If a problem should occur in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem. See the appropriate VTAM documentation for explanations of the return codes.

EXX4037 APPCCMD(CONTROL/REJECT).

Explanation: This is an informative message usually found only in the Expedite Base/MVS APPC trace. It reflects the result of a VTAM APPCCMD macro with CONTROL=CONTROL and QUALIFY=REJECT.

User Response: If a problem should occur in the communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem. See the appropriate VTAM documentation for explanations of the return codes.

EXX4038 TP_id() Con_id().

Explanation: This message provides information about the current conversation.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4039 IE Common Front End connection timer pop.

Explanation: This message indicates that the time limit has been exceeded for data coming from the Information Exchange Common Front End.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4040 APPCCMD(SEND/DATAFLU).

Explanation: This message reflects the results of a VTAM APPCCMD macro with CONTROL=SEND and QUALIFY=DATAFLU.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4041 APPCCMD(RECV/SPEC).

Explanation: This message reflects the results of a VTAM APPCCMD macro with CONTROL=RECV and QUALIFY=SPEC.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4042 APPCCMD(RECV/FMH7).

Explanation: This message reflects the results of a VTAM APPCCMD macro with CONTROL=RECV and QUALIFY=FMH7.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4043 FMH7 product set ID.

Explanation: This message documents the product set ID received on FMH-7 from the partner LU (Information Exchange).

User Response: Contact Help desk for help in resolving this problem.

EXX4044 FMH7 received wo/product set ID subvector.

Explanation: An FMH-7 received from the partner LU (Information Exchange) did not have a product set ID.

User Response: Contact Help desk for help in resolving this problem.

EXX4045 RPL6RCPR(), RPL6RCSC().

Explanation: This message indicates information about the current conversation.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4046 Conv_subtask_TCB at ().

Explanation: This message indicates information about the current conversation.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4047 Open macro R15().

Explanation: The result of an MVS OPEN macro was non-zero. This usually indicates a failure within MVS.

User Response: Contact Help desk for help in resolving this problem.

EXX4048 LcMainCb() call failed.

Explanation: A call to locate the main control block failed. This indicates a failure within Expedite Base/MVS.

User Response: Contact Help desk.

EXX4049 Conv_subtask abended.

Explanation: One of the subtasks used for APPC communication has abended. This may indicate an error within Expedite Base/MVS.

User Response: Contact Help desk.

EXX4050 APPCCMD(SEND/DATA).

Explanation: This message reflects the results of a VTAM APPCCMD macro with CONTROL=SEND and QUALIFY=DATA.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4051 Last data sent was ().

Explanation: When an error occurs in the APPC communication portion of Expedite Base/MVS, for diagnostic purposes, the last block of data that was sent to Information Exchange is displayed in this message.

User Response: None.

EXX4052 APPCCMD(DEALLOC/ABNDPROG).

Explanation: This message reflects the results of a VTAM APPCCMD macro with CONTROL=DEALLOC and QUALIFY=ABNDPROG.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4053 Expedite Base/MVS Version xxxxxx/yyyyyyyy.

Explanation: This message indicates the version of Expedite Base/MVS that is being executed. It is an informational message that is written to the job log to aid in the diagnosis of the problem.

User Response: None.

EXX4054 VTAM Attn exit: function = ???.

Explanation: This message indicates that a VTAM attention exit has been driven; however, the function is unknown. This could indicate an error in VTAM.

User Response: Contact Help desk for help with this problem.

EXX4055 Main CB not found.

Explanation: The main control block for APPC communications processing cannot be found. This may indicate a failure in MVS.

User Response: Contact Help desk for help with this problem.

EXX4056 XX must have LRECL>=121.

Explanation: The dataset in which trace messages are to be written must have an LRECL of > = 121 bytes and < = 255 bytes.

User Response: Please reallocate the dataset to these characteristics.

EXX4057 Conv_CB at.

Explanation: This message indicates information about the current conversation.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/ MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4058 (LOSS).

Explanation: This message indicates the LOSS exit has been driven and that Expedite Base/ MVS is no longer in session with Information Exchange.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/ MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX4059 TCP/IP error X, Y.

Explanation: This message indicates an invalid return code during TCP/IP processing. X indicates the error number. Y is a short description of the error.

User Response: If a problem occurs in the TCP/IP communication portion of Expedite Base/ MVS, this message is written to the job log to aid in diagnosis of the problem.

EXX4060 APPCCMD Failure: Control = X Qualify = Y R15(Z) R0(V).

Explanation: This message reflects the results of a VTAM APPCCMD macro. X indicates the control parameter on the failing command. Y indicates the QUALIFY parameter on the failing command. Z represents the decimal value returned in register 15. V represents the decimal value returned in register 0.

User Response: If a problem occurs in the APPC communication portion of Expedite Base/MVS, this message may be written to the job log to aid in diagnosis of the problem.

EXX5000 ERROR OCCURRED ISSUING "ESTAE" MACRO.

Explanation: This message is issued by the compression routines.

User Response: Contact bTrade, Inc. at 800-425-0444 for help in correcting this problem.

EXX5001 ERROR OCCURRED, BUT OUTMSGC OPEN FAILED.

Explanation: This message is issued by the compression routines and indicates a problem opening OUTMSGC.

User Response: Verify that OUTMSGC is a valid data set with valid characteristics. If the problem persists, contact bTrade at 800-425-0444.

Using data compression

Expedite Base/MVS provides integrated data compression and decompression through components of the TDAccess product. These components, formerly known as Comm-Press, may not be available in all countries. References in this manual will refer to these compression and decompression routines as Comm-Press.

Compression reduces the size of the files that are transmitted through Information Exchange. Significant savings in network charges and transmission time are possible when using data compression supplied by Comm-Press, Inc. Additional information about TDAccess can be found at: www.clickcommerce.com.

When you use data compression, some Expedite Base/MVS parameters are impacted, and session restarts may have to be handled differently. These considerations are described in this appendix.

NOTE: When using the Comm-Press compression and decompression routines, both the sender and receiver of compressed data must have the licensed Comm-Press product in order to compress and decompress the data.

Understanding the Comm-Press files used with Expedite Base/MVS

You need the following modules to use the Comm-Press product:

- INMSGP
  This compression program reads the INMSG file, compresses the appropriate files based on the COMPRESS parameter, and builds an INMSGC file reflecting the location of the compressed files for transmission. (Provided in TDAccess)
- OUTMSGP
  This decompression program reads the OUTMSG file after Expedite Base/MVS receives the files, then decompresses the received compressed files. (Provided in TDAccess)
- EXXERCMP
  This file contains the text of the error messages issued when the Comm-Press modules encounter an error. This file is sent on the product tape with Expedite Base/MVS. Your systems programmer can provide you with the file name to use.

Preparing work files for data compression

When using compression, OUTMSG must be a sequential DASD data set. OUTMSG cannot be a SYSOUT data set or a temporary data set.

Additional files are required. You must create the following work files when using Expedite Base/MVS with the Comm-Press product:

- COMPPDS DD file
- COMPWRK DD file
- CPLOOKUP DD file (required only when using COMPRESS(T) or COMPRESS(V))
- INMSGC DD file
- INMSGR DD file
- OUTMSGR DD file

All files except COMPPDS and OUTMSGC must be sequential data sets on direct access storage (DASD). DISP=MOD is not permitted. COMPPDS must be a partitioned data set. OUTMSGC can be a SYSOUT data set.

COMPPDS

INMSGP writes compressed files into this partitioned data set (PDS). EXXBASER reads from this file when transmitting the files listed on SEND and SENDEDI commands for which compression has been specified. You should not delete, move, or rename these members. You should instead complete the Information Exchange session or allow RESET processing to remove the temporary compressed members. If the same PDS is used repeatedly, you will need to periodically compress the PDS with IEBCOPY.

Create COMPPDS with these attributes:

This attribute:	Has these characteristics:
Space	Varies, depending on the amount of data you transmit. This file must be large enough to hold all of the compressed data you plan to send during an Expedite Base/MVS session.
Record format	Can have fixed or variable records. Variable records are usually more efficient.
Record length	Must be at least 72 bytes, but no more than 255 bytes.
Data set organization	Must be a partitioned data set and contain enough directory blocks to hold a separate member for each compressed file.

COMPWRK

OUTMSGP uses this work file during decompression. When compressed data is received, it is decompressed into the COMPWRK file. OUTMSGP then copies the data back into the received data set specified on the RECEIVE or RECEIVEEDI command. Although COMPWRK is a work file, it must be a cataloged data set.

Create COMPWRK with these attributes:

This attribute:	Has these characteristics:
Space	Varies, depending on the amount of compressed data you receive. This file must be large enough to hold the largest decompressed file you plan to receive during an Expedite Base/MVS session.

Record format	Not applicable (dynamically allocated by OUTMSGP).
Record length	Not applicable (dynamically allocated by OUTMSGP).

CPLOOKUP

This is the compression lookup table. It specifies whether compression should be performed for a particular sender/receiver pair.

Create CPLOOKUP with fixed-length, 80-byte records. For the format of this file, see “Compressing files with COMPRESS(T) or COMPRESS(V)” on page 303.

INMSGC

INMSGP writes to this file. INMSGC contains all the entries found in INMSG, plus the name and location of the corresponding compressed files. EXXBASER reads INMSGC to determine what commands to process. INMSGC is set to empty upon successful completion of Expedite Base processing. However, if you have specified a data recovery method other than session-level recovery and your Information Exchange session does not complete successfully, INMSGC is not set to empty and must remain unchanged for successful data recovery processing.

If the file is allocated as DISP=NEW, the DCB values should not be specified in the JCL. INMSGP will copy the DCB values from INMSG.

Create INMSGC with these attributes:

This attribute:	Has these characteristics:
Space	No minimum space requirement. This file must be large enough to hold all of your message commands.
Record format	Must be the same as the INMSG format.
Record length	Must be the same as the INMSG length.

INMSGR

This file is used in recovery situations to synchronize INMSG and INMSGC.

If the file is allocated as DISP=NEW, the DCB values should not be specified in the JCL. INMSGP will copy the DCB values from INMSG. Create INMSGR with these attributes:

This attribute:	Has these characteristics:
Space	No minimum space requirement. This file must be large enough to hold all of your message commands.
Record format	Must be the same as the INMSG format
Record length	Must be the same as the INMSG length.

OUTMSGC

INMSGP and OUTMSGP write compression and decompression errors to this file. You can allocate OUTMSGC as a SYSOUT data set.

Create OUTMSGC with these attributes:

This attribute:	Has these characteristics:
Space	Must be large enough to hold all of the compression and decom-pression error messages.
Record format	Must be fixed length.
Record length	Must be 80 bytes.

OUTMSGR

OUTMSGP uses this file as a work file during decompression. All the messages in OUTMSG are copied to OUTMSGR as the first step in the decompression process. The messages are updated and copied back to OUTMSG as the decompression takes place.

Although OUTMSGR is a work file, you should not allocate it as a temporary data set. This assists the restart and recovery process.

Create OUTMSGR with these attributes:

This attribute:	Has these characteristics:
Space	Must be large enough to hold all of the data in OUTMSG.
Record format	Must be the same as the OUTMSG format.
Record length	Must be the same as the OUTMSG length.

Sample JCL for data compression

You can use the following sample JCL to run an IEBASE job with the lookup table included instream. This sample JCL includes the additional DD statements that are required for data compression and decompression. The additional DD statements are highlighted.

```
//EXP EXEC PGM=IEBASE,REGION=0K
//STEPLIB DD DISP=SHR,DSN=user01.iebase.loadlib
// DD DISP=SHR,DSN=user01.compress.loadlib
//INPRO DD *

IDENTIFY IEACCOUNT(ieacct) IEUSERID(ieuserid) IEPASSWORD(iepassword);
SNACOMM IELUNAME(ieluname) USERLUNAME(userluname);

//INMSG DD DISP=SHR,DSN=user01.iebase.inmsg(file)

//CPLOOKUP DD *

SENDER(acct userid) RECEIVER(acct userid) COMPRESS(Y);

SENDER(acct userid) RECEIVER(acct userid) COMPRESS(Y);

SENDER(acct userid) RECEIVER(acct userid) COMPRESS(Y);

//INMSGR DD DSN=&&INMSGR,DISP=(NEW),
// UNIT=SYSDA,SPACE=(TRK,(5,1)),
// LRECL=80,BLKSIZE=0,RECFM=FB
//INMSGC DD DSN=user01.inmsgc,DISP=(NEW,CATLG),
// UNIT=SYSDA,SPACE=(TRK,(5,1)),
// LRECL=80,BLKSIZE=0,RECFM=FB

//OUTMSG DD DSN=user01.outmsg,
// DISP=(NEW,CATLG),UNIT=SYSDA,
// SPACE=(TRK,(1,1)),
// LRECL=80,BLKSIZE=0,RECFM=FB
```

```
//OUTMSGR DD DSN=user01.outmsgr,DISP=(NEW,CATLG),
// UNIT=SYSDA,SPACE=(TRK,(5,1)),
// LRECL=80,BLKSIZE=0,RECFM=FB
//OUTMSGC DD SYSOUT=*
//COMPPDS DD DSN=user01.compppds,
// DISP=(NEW,CATLG),UNIT=SYSDA,
// SPACE=(CYL,(1,1,10)),
// LRECL=80,BLKSIZE=0,RECFM=FB
//COMPWRK DD DSN=user01.compwrk,
// DISP=(NEW,CATLG),UNIT=SYSDA,
// SPACE=(CYL,(1,1))
```

```
//BASETRC DD SYSOUT=*
//ERRORMSG DD DISP=SHR,DSN=exxhlq.HQK6460.SEXXMSGS(EXXMSG)
//ERRORTXT DD DISP=SHR,,DSN=exxhlq.HQK6460.SEXXMSGS(EXXTXT)
//OUTPRO DD SYSOUT=*
//ERRORCMP DD DISP=SHR,DSN=exxhlq.HQK6460.SEXXMSGS(EXXERCMP)
```

NOTE: OUTMSG cannot be specified as SYSOUT.

Compressing files with COMPRESS(Y)

When COMPRESS(Y) is included on the SEND or SENDEDI command and INMSGP is available, the specified file is compressed prior to transmission. If the file to be sent contains multiple EDI envelopes, all envelopes are sent compressed. Additional parameters appear in the OUTMSG file along with the echoed SEND command:

```
COMSW(COMMPRESS) COMVER(3.1) COMFILE(nnnnn.nnnn(nnnnnn))
```

Where COMSW refers to the software providing the compression, COMVER refers to the version and release of that software, and COMFILE refers to the file name for the compressed file to be sent. This information is also contained in the CDH, to allow proper decompression at the receiving locations.

Compressing files with COMPRESS(T) or COMPRESS(V)

When COMPRESS(T) or COMPRESS(V) is included on the SEND or SENDEDI command, the specified file is compressed prior to transmission only if the SENDER, RECEIVER, and COMPRESS parameters are listed in the CPLOOKUP file. This file defines a series of paired receivers and senders, and for each pair indicates whether compression should be performed.

Each entry in the compression lookup table must follow this format:

```
    sender(sender) receiver(receiver) compress(y|n);
```

sender
Indicates the account and user ID or EDI source of a sender.

receiver
Indicates the account and user ID, alias and aliasname, listname, or EDI destination of a receiver.

compress
Indicates whether compression should be performed for this sender/receiver pair.

y	Compress the data for this sender/receiver pair.
n	Do not compress the data for this sender/receiver pair.

The following is an example of a compression lookup table:

```
SENDER(acct1 user01) RECEIVER(acct1 user02)  COMPRESS(y);
SENDER(acct1 user01) RECEIVER(alias1 alias2) COMPRESS(y);
SENDER(acct1 user01) RECEIVER(acct1 user03)  COMPRESS(y);
SENDER(acct1 user01) RECEIVER(listname02)    COMPRESS(y);
```

For each SEND command, INMSGP identifies the sender from a START command in INMSG or from an IDENTIFY command if AUTOSTART(Y) is specified in INPRO. INMSGP identifies the receiver from the ACCT and USERID, ALIAS and ALIASNAME, or LISTNAME parameters of the SEND command.

When entering values for the sender and receiver parameters, the ACCT or ALIAS must be 7 characters long (padded with blanks, if necessary). The USERID or ALIASNAME must begin in the next character position.

For the SENDEDI command, INMSGP identifies the sender and receiver from the EDI header. INMSGP looks for a corresponding entry in the CPLOOKUP file. Comm-Press supports X12, UCS, EDIFACT, and UN/TDI EDI formats. The sender and receiver entries for EDI data must match exactly what appears in the appropriate field of the EDI header. The inmsgp program examines only the EDI header to resolve sender/receiver pairs. The Expedite EDI tables are not used. For a description of which EDI header fields are used to identify the EDI source and destination, see Chapter 4, "Sending and receiving EDI data."

The COMPRESS parameter and the CPLOOKUP file allow you to control what gets compressed, based on the receiver. The CPLOOKUP file, like the INPRO and INMSG files, can be edited and modified when IEBASE is not running. The JCL example in the previous section of this appendix is an instream example of the CPLOOKUP file.

NOTES:

1. Specify COMPRESS(T) when you want to receive a warning message in the message response file, OUTMSG, if the sender/receiver pair is not in the CPLOOKUP file.

2. Specify COMPRESS(V) when you do not want to receive a warning message in the message response file, OUTMSG, if the sender/receiver pair is not in the CPLOOKUP file.

Decompressing received compressed files

When compressed files are received, the OUTMSG file contains new parameters used by the OUTMSGP program to process the received compressed files. The OUTMSGP program reads the OUTMSG file and decompresses the data. The following new parameters are found in the OUTMSG file for each compressed file received:

```
COMSW(COMM-PRESS) COMVER(3.1)
COMFILE(nnnn.nnnn(nnnn)) DCMPRC(.....)
```

All of the parameters except DCMPRC are obtained from the CDH. When the received file is not compressed, none of the data compression parameters shown above appear in the OUTMSG entry.

The OUTMSGP program copies the OUTMSG file to the OUTMSGR file, then processes and copies each response record back into OUTMSG. When a RECEIVED response record indicates compressed data, OUTMSGP decompresses the data in the received file into the COMPWRK file. Any uncompressed messages contained in the received file are also copied to the COMPWRK file. After successful decompression, all data is copied from COMPWRK back into the received data set.

The DCMPRC parameter in the RECEIVED response record provides a Comm-Press return code. You can use this return code to verify successful decompression processing or determine that an error condition exists. The DCMPRC parameter value of “00000” indicates successful decompression processing. If DCMPRC is not “00000”, see the related error messages at the end of this appendix to determine the appropriate action. Error messages are also written to the OUTMSGC file.

Decompressing RECEIVESTREAM data

In some situations, you may want to decompress files directly, instead of through Expedite Base/MVS. For example, you may be decompressing data from the OUTMSG file that was received as the result of a RECEIVESTREAM command. The DECOMP program is provided for these situations. The DECOMP program is included with TDAccess.

The following sample JCL provides an example for decompressing data in the OUTMSG file after the RECEIVESTREAM command was used to receive files from Information Exchange:

```
//DECOMP EXEC PGM=DECOMP,REGION=0K
//STEPLIB DD DISP=SHR,DSN=user01.load.library
//SYSPRINT DD SYSOUT=*
//DATAIN DD DISP=SHR,DSN=user01.outmsg.file
//DATAOT DD DSN=user01.outmsg.decomp,DISP=(NEW,CATLG),
// UNIT=SYSDA,SPACE=(CYL,(1,1)),
// DCB=user01.outmsg.file
```

NOTE: PARM='EDI' is needed on the EXEC statement when decompressing EDI-formatted data.

Expedite Base/MVS considerations when using the COMPRESS parameter

Most of the command parameters, when used with the COMPRESS parameter, are supported as documented in this publication. However, the following command parameters function differently when they are used with the compression parameters.

DATATYPE(E|B) on SEND commands
INMSGP uses the DATATYPE parameter during the compression process to determine whether ASCII or EBCDIC translation should be performed. After compression, the data is sent with DATATYPE(B).

DELIMITED(C|N|L) on SEND commands
INMSGP uses the DELIMITED parameter during the compression process to determine whether the data is delimited with carriage-return or line-feed characters. After compression, the data is sent with DELIMITED(N).

ENDSTR(endstring) on SENDSTREAM and RECEIVESTREAM commands
This parameter is permitted only when the endstring is on a record boundary.

RESRECL(S|E) on RECEIVE commands
Decompressed data is always treated as if RESRECL(S) was specified (that is, records are split if they are too long).

FORMAT(Y|N)
The FORMAT parameter is not supported, and causes an error if specified.

DLMOVERRIDE(Y|N)
The DLMOVERRIDE parameter is not supported.

EDIOPT(F)
The EDIOPT(F) parameter is not supported.

NOTE: For EDI data sent using the SENDEDI command, the character used as the segment terminator must not be in the range X'E0' to X'FF'.

Restart and recovery considerations with Comm-Press

The logic used in restart and recovery situations, described earlier in this publication, applies to restart and recovery situations where some or all of the files sent and received are compressed.

Because INMSGP processes the INMSG file before Expedite Base/MVS processing takes place, INMSGP must determine whether a restart situation exists before it does any processing. If the INMSGC file is not empty, INMSGP assumes that a restart is required.

INMSGP restarts by comparing the INMSG and OUTMSG files. All commands that were echoed to the OUTMSG file are left unchanged in the INMSGC file and are not reprocessed. Remaining commands in the INMSG file are processed (that is, any requested compression is performed) and copied to the INMSGC file.

Because of the way INMSGP uses the INMSGC file to determine restart status, you should be careful to avoid using an old INMSGC data set that might contain data. Unpredictable results would occur. You may want to use the RESET runtime parameter each time you run IEBASE, except for known restart situations. The RESET parameter causes all INMSG commands to be processed.

A special recovery situation can occur during decompression. After received data is decompressed, it is copied from the COMPWRK file back to the received file. If the received file fills up, an error message is written to the OUTMSGC file and processing stops. To recover from this situation, take the following steps:

1. Allocate a larger file.
2. Copy the data out of COMPWRK into the new file. (You could use IEBGENER to do this.)
3. Rerun IEBASE using the DECOMP runtime parameter to complete OUTMSGP processing.

Failure to properly dispose of the data in the COMPWRK file prior to rerunning IEBASE could result in the loss of the received data.

Restarting the session when using the COMPRESS parameter: In restart situations, you cannot change certain Expedite Base/MVS files, such as the INMSG and OUTMSG files. You also cannot change the INMSGC file. The program uses the INMSGC file when it processes compressed files.

Restarting the session after modifying INMSG: As noted earlier in this publication, you can modify entries in INMSG that are in error and restart the session. However, you cannot modify INMSG entries that have already been echoed to the OUTMSG file.

Using IEBASE RESET: When you use IEBASE RESET, you must remove the completed SEND and RECEIVE requests from the INMSG file.

Restarting the session for OUTMSGP processing: When the DCMPRC parameter indicates an error condition, you may be able to correct the error condition and restart the OUTMSGP processing to decompress received files. To restart just the OUTMSGP decompression processing, specify the DECOMP parameter with IEBASE as follows:

```
//EXP EXEC PGM=IEBASE,REGION=0K,PARM='DECOMP'
```

Decompressing files independently of Expedite Base/MVS: In some error recovery situations, you may want to decompress files directly, instead of through Expedite Base/MVS. Use the DECOMP program to do this, as explained in “Decompressing RECEIVESTREAM data” on page 305.

Error messages and return codes for data compression

The INMSGP and OUTMSGP modules write error messages to OUTMSGC for error conditions that require user intervention. The modules set the MVS completion code to 8. OUTMSGP also updates the DCMPRC field in the RECEIVED response record to indicate the results of decompressing that received message.

Some decompression errors do not prevent further processing, and only result in a non-zero value being placed in the DCMPRC field of the appropriate RECEIVED response record. For example, assume that a compressed message is corrupted during transmission. The cyclic-redundancy check (CRC) would fail, resulting in a DCMPRC return code of 26015. Processing would continue with the next RECEIVED response record. And, if no other session errors occurred, the SESSIONEND return code would be changed to 28010, indicating that not all commands were processed successfully.

NOTE: The compression code was formerly know as Comm-Press. Click Commerce, Inc. now owns and supports this code. Error messages reference the original name and ownership.

26001 Error allocating memory.

Explanation: A request for storage failed.

User Response: Specify a minimum region size of 4096K. See Restart and Recovery Considerations with Comm-Press for more information.

26003 Invalid use of COMPRESS parameter.

Explanation: COMPRESS(Y) or COMPRESS(T) was specified on a SEND or SENDEDI command, but the Comm-Press data compression software was not found.

User Response: Contact your marketing representative to acquire the Comm-Press data compression software.

26004 Unable to decompress received files.

Explanation: Compressed files were received, but the Comm-Press data compression software was not found.

User Response: Contact your marketing representative to acquire the Comm-Press data compression software.

26005 Invalid value for COMPRESS parameter.

Explanation: An invalid value was specified for the COMPRESS parameter.

User Response: Valid values are E, N, Y, T, and V. Consult the Expedite Base and Comm-Press user guides for instructions regarding the use of the COMPRESS parameter.

26006 Format parameter not valid with COMPRESS(Y).

Explanation: The FORMAT and COMPRESS parameters were both specified on a SEND command.

User Response: FORMAT is not supported when using compression. Remove one of the parameters.

26007 Cannot compress for reserved message class.

Explanation: An invalid message class was specified.

User Response: The message class is a reserved class for use by the STEDI system. These files cannot be compressed.

26008 TRANSLATE parameter not valid with COMPRESS(Y).

Explanation: The TRANSLATE and COMPRESS parameters were both specified on a SEND or SENDEDI command.

User Response: TRANSLATE is not supported when using compression. Remove one of the parameters.

26009 Compressed segment not found in file.

Explanation: The RECEIVED response record indicates compressed data was received; however, no compressed data was found in the received file.

User Response: The file has probably been corrupted during transmission. Receive the file again.

26011 Error freeing memory.

Explanation: A request to free storage failed.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26012 Trial period has expired.

Explanation: The Comm-Press trial period has expired.

User Response: Contact Comm-Press at 800-425-0444 to obtain a permanent software license.

26014 Compressed data ended prematurely.

Explanation: The compressed data has ended prematurely.

User Response: The data was probably truncated during transmission. The data has been removed from the received file and must be received again.

26015 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26017 Restricted license violation.

Explanation: You have a restricted version of the software that is limited to use with a certain trading partner.

User Response: Contact your marketing representative to obtain an unrestricted license for the Comm-Press data compression software.

26018 Invalid EDI envelope.

Explanation: The EDI header or trailer does not conform to the EDI standard, or a premature end-of-file condition was encountered on input. This error can also be caused by an invalid segment terminator in the EDI header.

User Response: Verify valid EDI headers and trailers are present. The segment terminator must be less than X'EO.'

26020 Error opening input file.

Explanation: An error occurred opening the file for compression or decompression.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26021 Error reading input file.

Explanation: An error occurred reading the file during compression or decompression.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26022 Error writing received file.

Explanation: A write error occurred copying the decompressed data from COMPWRK to the received file.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26023 Error closing input file.

Explanation: An error occurred closing the file after compression or decompression.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26024 Input file has an invalid record format.

Explanation: Only the fixed and variable record formats, blocked or unblocked, are supported. The undefined record format is not supported.

User Response: Verify the SEND and RECEIVE files conform to the record format restriction.

26030 Error opening COMPPDS/COMPWRK/RECEIVE file.

Explanation: An error occurred opening the file for compression or decompression.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26031 Error reading COMPWRK file.

Explanation: A read error occurred copying the decompressed data from COMPWRK to the received file.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26032 Error writing COMPPDS/COMPWRK/RECEIVE file.

Explanation: An error occurred writing the file during compression or decompression.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26033 Error closing COMPPDS/COMPWRK/RECEIVE file.

Explanation: An error occurred closing the file after compression or decompression.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26034 COMPPDS file has an invalid record format.

Explanation: Only the fixed and variable record formats, blocked or unblocked, are supported. The undefined record format is not supported.

User Response: Verify the COMPPDS file conforms to the record format restrictions.

26035 Error stowing COMPPDS file.

Explanation: An error occurred while saving a compressed member in the COMPPDS file.

User Response: This error probably occurred because the PDS is full. Examine the MVS system message to determine the cause of the error and rerun IEBASE.

26036 COMPPDS file has an invalid record length.

Explanation: COMPPDS must have a minimum fixed record length of 24 bytes.

User Response: Correct your JCL or COMPPDS file and rerun IEBASE.

26037 COMPPDS file has an invalid record length.

Explanation: COMPPDS must have a minimum variable record length of 28 bytes.

User Response: Correct your JCL or COMPPDS file and rerun IEBASE.

26050 Error opening KEYIN file.

Explanation: An error occurred opening the file containing the encryption key.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26051 Error reading KEYIN file.

Explanation: An error occurred reading the file containing the encryption key.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26054 Premature EOF on KEYIN file.

Explanation: The KEYIN file is empty.

User Response: A record containing the encryption key is required. Provide the encryption key and rerun IEBASE. See Restart and Recovery Considerations with Comm-Press for more information.

26055 Error decrypting received file.

Explanation: The received file did not decrypt successfully.

User Response: This is most likely due to an incorrect decryption key specified in the KEYIN file. Correct the key and rerun IEBASE. See Restart and Recovery Considerations with Comm-Press for more information.

26060 Error opening file.

Explanation: An error occurred opening the indicated file.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26061 Error reading file.

Explanation: An error occurred reading the indicated file.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26062 Error writing file.

Explanation: An error occurred writing the indicated file.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26063 Error closing file.

Explanation: An error occurred closing the indicated file.

User Response: Examine the MVS system message to determine the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26064 Invalid record format for file.

Explanation: Only the fixed and variable record formats, blocked or unblocked, are supported. The undefined record format is not supported.

User Response: Verify the indicated file conforms to the record format restrictions.

26065 INMSG and INMSGC files have mismatching DCB parms.

Explanation: INMSG and INMSGC must have the same record format and record length.

User Response: Correct your JCL and rerun IEBASE.

26091 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26092 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26093 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26094 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26095 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26096 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26097 Compressed segment in error.

Explanation: The error-checking routine indicated the compressed data was corrupted.

User Response: The data was probably corrupted during transmission. The data has been removed from the received file and must be received again.

26098 Invalid Comm-Press version.

Explanation: The received file was compressed with a newer version of the Comm-Press software.

User Response: Contact your marketing representative to acquire the latest version of the Comm-Press data compression software.

26200 Error allocating SEND/RECEIVE file.

Explanation: An error occurred during dynamic allocation of the SEND or RECEIVE file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26210 Error allocation COMPPDS/COMPWRK file.

Explanation: An error occurred during dynamic allocation of the COMPPDS or COMPWRK file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26220 Error allocating received file.

Explanation: An error occurred during dynamic allocation of the received file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26221 Error de-allocating received file.

Explanation: An error occurred during dynamic de-allocation of the received file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26222 Error retrieving COMPWRK data set name.

Explanation: An error occurred during dynamic allocation of the COMPWRK file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26223 Error allocating COMPWRK file.

Explanation: An error occurred during dynamic allocation of the COMPWRK file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26224 Error de-allocating COMPWRK file.

Explanation: An error occurred during dynamic de-allocation of the COMPWRK file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26225 Error allocating SEND file.

Explanation: An error occurred during dynamic allocation of the SEND file.

User Response: Examine the dynamic allocation error codes in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

26230 Error loading INMSGP module.

Explanation: An error occurred loading the INMSGP module.

User Response: Examine the abend reason code in the accompanying message. Contact your systems programmer for help in determining the cause of the error.

26231 Error loading OUTMSGP module.

Explanation: An error occurred loading the OUTMSGP module.

User Response: Examine the abend reason code in the accompanying message. Contact your systems programmer for help in determining the cause of the error. See Restart and Recovery Considerations with Comm-Press for more information.

Migrating from expEDIte/MVS Host

This chapter is written for those who currently use expEDIte/MVS Host, but are switching to Expedite Base/MVS. It contains a section for each function provided by expEDIte/MVS Host. These sections describe how to perform the same function in Expedite Base/MVS. You should read Chapter 1, “Introducing Expedite Base/MVS.” and Chapter 2, “Setting up files, including the JCL.” before reading this chapter.

“Comparison summary” on page 336 contains an example of an expEDIte/MVS Host job and an example of an Expedite Base/MVS job that perform the same functions.

Although they are not discussed in this chapter, return codes in Expedite Base/MVS have also changed. For full descriptions of all return codes, see Appendix D, “Expedite Base/MVS messages and codes.”

NOTE: Your VTAM definitions are not the same for the two products. See the Program Directory that accompanies the Expedite Base/MVS product tape for more information on VTAM definitions and related installation requirements.

Session-level recovery

expEDIte/MVS Host uses only session-level recovery, while Expedite Base/MVS offers additional types of data recovery (checkpoint-level, file-level, and user-specified). If you want to make your migration to Expedite Base/MVS as simple as possible, continue using session-level recovery. Session-level recovery is the default for Expedite Base/MVS, so you do not have to do anything to request it.

JCL comparison

This section tells you how the JCL used in Expedite Base/MVS is different from the JCL used in expEDIte/MVS Host.

EXEC statement

The program name of the EXEC statement (EXEC PGM) used for expEDIte/MVS Host is TPMAIN. For Expedite Base/MVS, use the program name IEBASE.

PARM option

The PARM option of the EXEC statement no longer requires the same parameters needed for expEDIte/MVS Host. The NOSPIE and NOSTAE options are no longer used. Some of the information you provided in the expEDIte/MVS Host PARM options you now specify in profile commands in the profile command file (ddname INPRO).

expEDIte/MVS Host has 10 positional parameters in the PARM option. These parameters are listed below with an explanation of how you specify the same information in Expedite Base/MVS.

userapplid
Use a new userapplid value for Expedite Base/MVS. The user-applid you used for expEDIte/MVS Host will not work for Expedite Base/MVS. Specify this new value in the USERLUNAME parameter of the SNACOMM profile command.

NOTE: USERLUNAME is a required parameter in Expedite Base/MVS.

ieapplid
Because Expedite Base/MVS uses LU 6.2 communication, you must use a different ieapplid value. Specify this value in the IELUNAME parameter of the SNACOMM profile command. The default IELUNAME value is ibm0rely.

unused
expEDIte/MVS Host does not use this positional parameter.

unused
expEDIte/MVS Host does not use this positional parameter.

unused
expEDIte/MVS Host does not use this positional parameter.

blksize
Do not specify this parameter in Expedite Base/MVS. Expedite Base/MVS always uses a block size of 12000 to maximize performance. This block size does not conflict with your network configuration.

indd
Specify this value with the INMSG PARM option of Expedite Base/MVS. For details on how to code this option, see “EXEC statement” on page 21.

outdd
Specify this value with the OUTMSG PARM option of Expedite Base/MVS. For details on how to code this option, see “EXEC statement” on page 21.

api
Do not specify this parameter in Expedite Base/MVS. Expedite Base/MVS supports only one API.

cpage
Specify this value in the CODEPAGE parameter of the IDENTIFY profile command.

Data definition names

You must use different data definition names (ddnames) for Expedite Base/MVS. This section summarizes ddname changes with which you should be familiar.

Changed ddnames

The following ddnames are changed in Expedite Base/MVS.

expEDIte/MVS Host ddname:	Expedite Base/MVS ddname:
INFILE	INMSG
OUTFILE	OUTMSG
QUTTABLE	QUALTBL

ddnames not used in Expedite Base/MVS

The following ddnames, used by expEDIte/MVS Host, do not exist in Expedite Base/MVS:

- PLIDUMP
- ISCMSGS
- ISCTRACE

New ddnames

The following session-level recovery ddnames are new for Expedite Base/MVS. They do not have counterparts in expEDIte/MVS Host.

This ddname:	Defines a file that:
INPRO	Contains your profile information, such as your Information Exchange account and user ID.
OUTPRO	Contains the responses generated by Expedite Base/MVS when processing the profile information provided in INPRO.
BASETRC	Holds trace information that GXS Community Support might require for problem determination.
LINKTRC	Holds trace information that GXS Community Support might require for problem determination.
ERRORMSG	Contains error messages that can be helpful to you if an error occurs. This file must be a partitioned data set.
ERRORTXT	Contains extra error description text that provides detailed information if an error occurs. This file must be a partitioned data set.
TRACEMSG	Contains trace message text with trace information.

There are also four new ddnames for checkpoint-level, file-level, and user-initiated recovery. (For more information, see “Setting up work files” on page 47.) These new ddnames are:

- OUTWORK
- RCVWORK
- SESSION
- EDIWORK

These ddnames define files that contain information internal to Expedite Base/MVS for managing data recovery.

EDI qualifier table

With expEDIte/MVS Host, you used the ddname QUTTABLE for the EDI qualifier table. With Expedite Base/MVS, use QUALTBL. The format of the EDI qualifier table has also changed. For more information, see “EDI qualifier table (QUTTABLE)” on page 323.

EDI destination tables

With Expedite Base/MVS, the EDI destination tables continue to have the following ddnames:

This data type:	Has this default ddname:
EDIFACT	TTABLExx, where *xx* is the ID qualifier of the EDI destination.
X12	TTABLExx, where *xx* is the ID qualifier of the EDI destination.
UCS	TTABLE01.
UN/TDI	IEUNTDI.

NOTE: The format of this table has changed. For more information, see “EDI destination table” on page 323.

SYSPRINT

Both expEDIte/MVS Host and Expedite Base/MVS write information messages to the SYSPRINT ddname. SYSPRINT is not required, but it is recommended that you specify a SYSPRINT.

Starting an Information Exchange session

With expEDIte/MVS Host, you must use a CSS command to start an Information Exchange session. Expedite Base/MVS automatically starts the Information Exchange session with the information you specify in the IDENTIFY profile command. If you do not want Expedite Base/ MVS to automatically start the Information Exchange session, use the AUTOSTART(n) parameter of the TRANSMIT profile command to stop the automatic session start. You must then place a START command in the message command file (INMSG) to start the Information Exchange session.

CSS command

This section tells you which Expedite Base/MVS parameters correspond to the old CSS command fields. Most of the Expedite Base/MVS parameters are part of the START command.

This CSS command field:	Corresponds to this parameter:
ACCTNO	ACCOUNT parameter of the START command.
USERID	USERID parameter of the START command.
PASSWORD	IEPASSWORD parameter of the START command.
NEWPSWRD	NIEPASSWORD parameter of the START command.

TIMEZONE	TIMEZONE parameter of the IDENTIFY profile command.
SYSTYPE	Expedite Base/MVS sets the system type to EXMV000T. Do not specify this information with Expedite Base/MVS.
SYSLEVEL	Expedite Base/MVS sets the system level to the current software level of the program. Do not specify this information with Expedite Base/MVS.
TESTREST	Expedite Base/MVS automatically chooses the appropriate value for this field. Do not specify this information with Expedite Base/MVS.

Session start comparison

Session start examples are provided here. Unlike expEDIte/MVS Host, Expedite Base/MVS requires you to enter message commands in one file (INMSG) and profile commands in another (INPRO).

expEDIte/MVS Host INFILE

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CSSACT1 USER01 MYPSWRD W0000 RESETSES
```

Expedite Base/MVS INMSG

```
START ACCOUNT(ACT1) USERID(USER01) PASSWORD(MYPSWRD);
```

Expedite Base/MVS INPRO

```
TRANSMIT AUTOSTART(N);
```

Ending an Information Exchange session

With expEDIte/MVS Host, you must use a CSE command to end an Information Exchange session. Expedite Base/MVS automatically ends the Information Exchange session after completing the last command in the message command file (INMSG). If you do not want Expedite Base/MVS to end the Information Exchange session automatically, use the AUTOEND(n) parameter of the TRANSMIT profile command to stop the automatic session end. You must then place an END command in the message command file (INMSG) to end the Information Exchange session.

Session end comparison

Session end examples are provided here. Unlike expEDIte/MVS Host, Expedite Base/MVS requires you to enter message commands in one file (INMSG) and profile commands in another (INPRO).

expEDIte/MVS Host INFILE

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CSE
```

Expedite Base/MVS INMSG

```
END;
```

Expedite Base/MVS INPRO

```
TRANSMIT AUTOEND(N);
```

Sending files

With expEDIte/MVS Host, you use the CSF command to send files and get an RSF in response. With Expedite Base/MVS, you use the SEND command and get a SENT record in response.

CSF command

This section tells you which SEND command parameters in Expedite Base/MVS correspond to the old CSF fields in expEDIte/MVS Host.

This CSF field:	Corresponds to this command parameter:
DESTTYP/DESTACCT/	SYSID/ACCOUNT/USERID, ALIAS/ALIASNAME or LISTNAME
DESTUID/DTBLTYP/ DTBLID	
MSGNCLS	MODE parameter of the SEND command.
MSGDCLS	PRIORITY parameter of the SEND command.
MSGCHRG	CHARGE parameter of the SEND command.
MSGRCPT	ACK parameter of the SEND command.
MSGNAME	MSGNAME parameter of the SEND command.
MSGSEQN	MSGSEQNO parameter of the SEND command.
INFORMAT	TRUNCATE parameter of the SEND command.
STGFORM	DELIMIT parameter of the SEND command.
MSGUCLS	CLASS parameter of the SEND command.
MSGSIZE	MSGSIZE parameter of the TRANSMIT profile command.
MSGRETN	RETAIN parameter of the SEND command.
MSGVCHK	VERIFY parameter of the SEND command.
MSGDTYPE	DATATYPE parameter of the SEND command.

CSF Part 2

The DATATYP and FILESPEC CSF Part 2 fields in expEDIte/MVS Host correspond to the FILEID parameter of the SEND command in Expedite Base/MVS.

D record

With expEDIte/MVS Host, you can add a description of the file by placing a D record after the CSF Part 2 command. With Expedite Base/MVS, you add this description with the DESCRIPTION parameter of the SEND command.

Send file comparison

Examples of commands used to send files are:

expEDIte/MVS Host

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CSFDACT1 USER002  3  CTESTCLAS
CSFAUSER01.TEST.DATA
```

Expedite Base/MVS INMSG

```
SEND FILEID(USER01.TEST.DATA) ACCOUNT(ACT1) USERID(USER002)
   CLASS(TESTCLAS);
```

RSF response record

The UNIQUEID RSF field in expEDIte/MVS Host corresponds to the UNIQUEID parameter in the Expedite Base/MVS SENT record.

Sending EDI files

With expEDIte/MVS Host, you use the CSP command to send EDI files and get an RSP in response to every EDI envelope sent with this command. With Expedite Base/MVS, you use the SENDEDI command and get a SENT record in response to every EDI envelope sent.

expEDIte/MVS Host command:	Expedite Base/MVS command
CSP command	SENDEDI command.
RSP record	SENT record.

CSP command

The following table shows which SEND parameters in Expedite Base/MVS correspond to the old CSP fields in expEDIte/MVS Host.

This CSP field:	Corresponds to this parameter:
MSGNCLS	MODE parameter of the SENDEDI command.
MSGCHRG	CHARGE parameter of the SENDEDI command.
MSGRCPT	ACK parameter of the SENDEDI command.
MSGNAME	MSGNAME parameter of the SENDEDI command.
MSGSEQN	MSGSEQNO parameter of the SENDEDI command.
MSGDCLS	PRIORITY parameter of the SENDEDI command.
MSGUCLS	CLASS parameter of the SENDEDI command.

MSGSIZE	MSGSIZE parameter of the TRANSMIT profile command.
MSGRETN	RETAIN parameter of the SENDEDI command.
MSGVCHK	VERIFY parameter of the SENDEDI command.

CSP Part 2

The following table shows which SENDEDI command parameters in Expedite Base/MVS correspond to the old CSP Part 2 fields.

This CSP Part 2 field or record:	Corresponds to this SENDEDI parameter:
DATATYP and FILESPEC	FILEID. Start the value with dd: to specify a ddname.
D record	DESCRIPTION

Send EDI comparison

Examples of expEDIte/MVS Host and Expedite Base/MVS command files for sending EDI data are:

expEDIte/MVS Host

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CSP 3 EDITEST

CSPAUSER01.EDI.TEST
```

Expedite Base/MVS

```
SENDEDI FILEID(USER01.EDI.TEST) CLASS(EDITEST);
```

RSP response record

The following table shows which SENT parameters in Expedite Base/MVS correspond to the old RSP fields in expEDIte/MVS Host. Note that all the SENT parameters listed here might not appear in every SENT record. Blank parameters do not appear. Also, mutually exclusive parameters, such as ACCOUNT and ALIASNAME, do not appear at the same time.

This RSP field or fields:	Corresponds to this SENT record parameter or parameters:
DESTTYP/DESTACCT/DESTUID/	SYSID/ACCOUNT/USERID,
DTBLTYP/DTBLID	ALIAS/ALIASNAME or LISTNAME
MSGUCLS	CLASS
MSGNAME	MSGNAME
MSGSEQN	MSGSEQNO
UNIQUEID	UNIQUEID
LENGTH	LENGTH

RSP Part 2 response record

The following table shows which SENT parameters in Expedite Base/MVS correspond to the old RSP Part 2 fields in expEDIte/MVS Host.

This RSP Part 2 field:	Corresponds to this SENT parameter:
EDITYPE	EDITYPE
EDIQUAL	QUALIFIER
EDIDEST	DESTINATION
CONTROL	CONTROLNUM

EDI qualifier table (QUTTABLE)

Expedite Base/MVS uses the ddname QUALTBL to refer to the EDI qualifier table. This section tells you which QUALTBL parameters in Expedite Base/MVS correspond to the old EDI qualifier table (QUTTABLE) fields in expEDIte/MVS Host.

This QUTTABLE field:	Corresponds to this QUALTBL parameter:
EDITYPE	DATATYPE
QUAL	QUALIFIER
D_ALIAS	ALIAS
DTABLE	TTABLE. Type **dd:** in front of the parameter value to specify a ddname.

Qualifier table comparison

Examples of commands that set up qualifier tables are:

expEDIte/MVS Host ddname QUTTABLE

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
X 01 GX01 TTABLE01
E 02 GE02 TTABLE02
```

Expedite Base/MVS ddname QUALTBL

```
DATATYPE(X) QUALIFIER(01) ALIAS(GX01) TTABLE(DD:TTABLE01);
DATATYPE(E) QUALIFIER(02) ALIAS(GE02) TTABLE(DD:TTABLE02);
```

EDI destination table

The following table shows which Expedite Base/MVS EDI destination table parameters correspond to the old expEDIte/MVS Host EDI destination table fields.

expEDIte/MVS Host EDI destination table field:	Corresponds to Expedite Base/MVS EDI destination parameter:
EDIDEST	EDIDEST
ACCOUNT	ACCOUNT

USERID	USERID
ALIASNAM	ALIASNAME
LISTNAME	LISTNAME
DESTTYPE	None. You do not need a parameter for this information with Expedite Base/MVS.
SYSID	SYSID
ALIAS	ALIAS

Destination table comparison

Examples of files used to set up destination tables are:

expEDIte/MVS Host (TTABLE01)

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
DUNS01 ACT2 USER001
DUNS02 ACT3 USER005
```

Expedite Base/MVS (TTABLE01)

```
EDIDEST(DUNS01) ACCOUNT(ACT2) USERID(USER001);
EDIDEST(DUNS02) ACCOUNT(ACT3) USERID(USER005);
```

Receiving files

With expEDIte/MVS Host, you use the CRF command to receive files, and you get RRF, RHI, RFL, RFN, and D records in response to this command. With Expedite Base/MVS, you use the RECEIVE command and get a single RECEIVED record in response.

CRF command

The following table shows which Expedite Base/MVS RECEIVE parameters correspond to the old CRF fields.

This CRF field:	Corresponds to this RECEIVE parameter:
SRCTYP/SRCACCT/ SRCUID/ DTBLTYP/ DTBLID	SYSID/ACCOUNT/USERID, ALIAS/ALIASNAME, or LISTNAME. If you want to receive only requeued archive messages, use the REQUEUED parameter. Do not specify any of these parameters if you want to receive all of your data, regardless of its source.
MSGUCLS	CLASS
ARCREF	ARCHIVEID
TYPRCV	ALLFILES
EDIOPT	EDIOPT
EDIPROC	AUTOEDI
STGFORMO	DLMOVERRIDE

CRF Part 2

The following table shows which Expedite Base/MVS RECEIVE parameters correspond to the old CRF Part 2 fields.

This CRF Part 2 field:	Corresponds to this RECEIVE command parameter:
STGFORM	DELIMITED
RESRECL	RESRECL
DATATYP and FILESPEC	FILEID. Start the file ID with dd: to specify a ddname.

NOTE: The DELIMITED parameter of the RECEIVE command of Expedite Base/MVS does not support the D value in the STGFORM option that was used in expEDIte/MVS Host. For information on how to send binary data, see Chapter 3, “Communicating with other operating systems.”

Receive file comparison

Examples of commands used to receive files are:

expEDIte/MVS Host

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CRFDACT1 USER002 TESTCLAS
CRFC AUSER01.TEST.OUT
```

Expedite Base/MVS

```
RECEIVE FILEID(USER01.TEST.OUT)  ACCOUNT(ACT1)
USERID(USER002) CLASS(TESTCLAS);
```

RRF response record

The following table shows which Expedite Base/MVS RECEIVED parameters correspond to the old RRF fields. Note that all the RECEIVED parameters listed here might not appear in every RECEIVED record. Blank parameters do not appear. Also, mutually exclusive parameters, such as ACCOUNT and ALIASNAME, do not appear together.

This RRF response record:	Corresponds to this RECEIVED parameter:
SRCTYP/SRCACCT/SRCUID/DTBLTYP/ DTBLID	SYSID/ACCOUNT/USERID, or
	ALIAS/ALIASNAME
MSGNAME	MSGNAME
MSGSEQN	MSGSEQNO
MSGNCLS	MODE
MSGUCLS	CLASS
MSGDCLS	PRIORITY
MSGRCPT	ACK

MSGCHRG	CHARGE
MSGSEQO	MSGSEQO
SYSTYPE	SYSNAME
SYSLEVEL	SYSLEVEL

RRF Part 2 response record

The following table shows which Expedite Base/MVS RECEIVED parameters correspond to the old RRF Part 2 fields used in expEDIte/MVS Host. All of the RECEIVED parameters listed here might not appear in every RECEIVED record. Parameters that are blank do not appear.

This RRF Part 2 field:	Corresponds to this RECEIVED parameter:
MSGDATE	MSGDATE
MSGTIME	MSGTIME
MSGTZONE	Because this field is always L, Expedite Base/MVS does not provide a parameter for it.
STGFORM	DELIMITED
RESRECL	If Expedite Base/MVS splits received output records, you get a WARNING record in the message response file. If Expedite Base/MVS ends because you specified RESRECL(e) on the RECEIVE command, you receive an error indicating that this occurred.
DATATYP and FILESPEC	FILEID parameter of the RECEIVED record.

D record (receiving)

This record corresponds to the DESCRIPTION parameter of the RECEIVED record. The DESCRIPTION parameter does not appear if the CDH does not contain a description.

RHI record

The following table shows which RECEIVED parameters in Expedite Base/MVS correspond to the old RHI records. Note that all the RECEIVED parameters listed here might not appear in every RECEIVED record. Parameters that are blank do not appear.

This RHI record:	Corresponds to this RECEIVED parameter:
RECORDF	RECFM
RLENGTH	RECLEN
DTYPE	DATATYPE
TTYPE	TRANSLATE
DELIMIT	RECDLM
DFORMAT	EDITYPE
UNIQUEID	UNIQUEID

CODEPAGE	CODEPAGE
SYSTYPE	SYSTYPE
VERSION	SYSVER
FILEDATE	FILEDATE
FILETIME	FILETIME

RFL record

This record corresponds to the SENDERLOC parameter of the RECEIVED record. This parameter does not appear if the SENDERLOC is blank or unavailable.

RFN record

This record corresponds to the SENDERFILE parameter of the RECEIVED record. This parameter does not appear if the SENDERFILE is blank or unavailable.

Receiving EDI files

With expEDIte/MVS Host, you use the CRP command to receive files, and RRP, RHI, RFL, RFN, and D records return in response to this command. With Expedite Base/MVS, you use the RECEIVEEDI command, and you get a single RECEIVED record in response.

NOTE: For the mapping of the RHI, RFL, RFN, and D records to the RECEIVED record, see “Receiving EDI files” on page 327.

CRP command

The following table shows which RECEIVEEDI parameters in Expedite Base/MVS correspond to the old CRP fields.

This CRP field:	Corresponds to this RECEIVEEDI parameter:
SRCTYP/SRCACCT/ SRCUID/ DTBLTYP/ DTBLID	SYSID/ACCOUNT/USERID, ALIAS/ALIASNAME, or LISTNAME. If you want to receive only requeued archive messages, use the REQUEUED parameter. Do not specify any of these parameters if you want to receive all of your data, regardless of its source.
MSGUCLS	CLASS
ARCREF	ARCHIVEID
TYPRCV	ALLFILES
EDIOPT	EDIOPT

CRP Part 2

Use the Expedite Base/MVS FILEID parameter to specify the same information the DATATYP and FILESPEC fields specify in expEDIte/MVS Host. Begin the file ID with dd: to specify a ddname.

Receive EDI file comparison

Examples of command files used to receive EDI files are shown below.

expEDIte/MVS Host

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CRPDACT1 USER002 EDITEST
CRPAUSER01.EDI.OUT
```

Expedite Base/MVS

```
RECEIVEEDI  FILEID(USER01.EDI.OUT)
ACCOUNT(ACT1) USERID(USER002)  CLASS(EDITEST);
```

RRP response record

The format of part 1 of the RRP record is identical to the format of part 1 of the RRF record. For more information, see “RRF response record” on page 325.

RRP Part 2 response record

The following table shows the RECEIVED parameters in Expedite Base/MVS that correspond to the old RRP Part 2 response record. All the RECEIVED parameters listed here might not appear in every RECEIVED record. Blank parameters do not appear.

This RRP Part 2 response record:	Corresponds to this RECEIVED parameter:
MSGDATE	MSGDATE
MSGTIME	MSGTIME
MSGTZONE	Because this field is always L, Expedite Base/MVS does not provide a parameter for it.
DATATYP and FILESPEC	FILEID parameter of the RECEIVED record.

Sending messages with T records

With expEDIte/MVS Host, you can send messages from INFILE using T records in combination with the CGH and CMH commands. With Expedite Base/MVS, you use the SENDSTREAM command to send data from INMSG.

With expEDIte/MVS Host, an RSM record returns for every message group that you send. With Expedite Base/MVS, a SENT record returns when the SENDSTREAM command is complete.

CGH command

The CGH command indicates the start of a message group to Information Exchange. You must have one (and only one) SENDSTREAM command for each CGH command. The following table shows the SENDSTREAM parameters in Expedite Base/MVS that correspond to the old CGH fields.

This CGH field:	Corresponds to this SENDSTREAM parameter:

DESTTYP/DESTACCT/DESTUID/	SYSID/ACCOUNT/USERID,
DTBLTYP/DTBLID	ALIAS/ALIASNAME, or LISTNAME
MSGNCLS	MODE
MSGUCLS	CLASS
MSGDCLS	PRIORITY
MSGCHRG	CHARGE
MSGRCPT	ACK
EORCHAR	ENDSTR
MSGRETN	RETAIN
MSGVCHK	VERIFY

CMH command

The CMH command indicates the start of a message to Information Exchange. With expEDIte/MVS Host, you are required to have at least one CMH command following a CGH command. With Expedite Base/MVS, you specify CMH fields in the SENDSTREAM command. The following table shows the SENDSTREAM parameters that correspond to the old CMH fields.

This CMH command:	Corresponds to this SENDSTREAM parameter:
MSGNAME	MSGNAME
MSGSEQN	MSGSEQNO

T records (sent)

With expEDIte/MVS Host, you placed your data in T records following a CMH record. With Expedite Base/MVS, you place your data immediately after the SENDSTREAM command. If necessary, you can begin your data on the record following the SENDSTREAM command. End your data with the string specified in the ENDSTR parameter.

Depending on the design of your application, you might have to make an adjustment, because SENDSTREAM does not require every data record to start with T.If you use fixed-length, 80-byte records, SENDSTREAM sends one more byte per record than it sends with T records. Use INMSG with fixed-length, 79-byte records to send 79 bytes per record (as you did with T records).

F record

With expEDIte/MVS Host, you can use an F record to specify the name of a file to be sent instead of specifying the actual data in T records. With Expedite Base/MVS, you must use the SEND command to send a file.

D record

With expEDIte/MVS Host, you can add a description of the file by placing a D record after the CGH command. With Expedite Base/MVS, you add this description with the DESCRIPTION parameter of the SENDSTREAM command.

RSM response record

The UNIQUEID parameter of the SENT command in Expedite Base/MVS corresponds to the UNIQUEID RSM field in expEDIte/MVS Host.

T record send comparison

Examples of commands used to send T records are:

expEDIte/MVS Host

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CGHDACT1 USER002 TESTCLAS 3
CMHTESTNAME
TLINE ONE OF THE DATA
TLINE TWO OF THE DATA
CSE
```

Expedite Base/MVS INMSG

```
SENDSTREAM ACCOUNT(ACT1) USERID(USER002) MSGNAME(TESTNAME)
CLASS(TESTCLAS) ENDSTR(XXX);
LINE ONE OF DATA
LINE TWO OF DATA
XXX
```

Receiving messages with T records

With expEDIte/MVS Host, you can use the CRM command to receive messages to OUTFILE using T records. An RGH record returns for every message group you receive, and an RMH record returns for every message you receive. You also get RHI, RFN, RFL, and D records for each message group you receive. For more information on these record, see "Receiving files" on page 324s.

With Expedite Base/MVS, you use the RECEIVESTREAM command to receive data to the message response file, OUTMSG, and a RECEIVED record returns for each message group received.

CRM command

The following table shows which RECEIVESTREAM parameters in Expedite Base/MVS correspond to the old CRM fields.

This CRM command field:	Corresponds to this RECEIVESTREAM parameter:
SRCTYP/SRCACCT/ SRCUID/ DTBLTYP/ DTBLID	SYSID/ACCOUNT/USERID, ALIAS/ALIASNAME, or LISTNAME parameters, as needed to specify this information. If you want to receive only requeued archive messages, use the REQUEUED parameter. If you want to receive all your data regardless of its source, do not specify any of these parameters.
MSGUCLS	CLASS
ARCREF	ARCHIVEID

EOMIND	ENDSTR
TYPRCV	ALLFILES

RGH record

The RGH record represents the receipt of a message group. The following table shows which Expedite Base/MVS RECEIVED parameters correspond to the RGH fields in expEDIte/MVS Host. Note that all of the RECEIVED parameters listed here might not appear in every RECEIVED record. Blank parameters do not appear. Also, mutually exclusive parameters, such as ACCOUNT and ALIASNAME, do not appear together.

This RGH record field:	Corresponds to this RECEIVED parameter:
SRCTYP/SRCACCT/SRCUID/DTBLTYP/ DTBLID	SYSID/ACCOUNT/USERID or
	ALIAS/ALIASNAME
MSGNCLS	MODE
MSGUCLS	CLASS
MSGDCLS	PRIORITY
MSGRCPT	ACK
MSGCHRG	CHARGE
SYSTYPE	SYSTYPE
SYSLEVEL	SYSVER

RMH record

With expEDIte/MVS Host, an RMH record returns for each message received within a message group. With Expedite Base/MVS, the RMH information is included in the RECEIVED parameters. The following table shows which Expedite Base/MVS RECEIVED parameters correspond to the old RMH record fields. All the RECEIVED parameters listed here might not appear in every RECEIVED record. Blank parameters do not appear.

This RMH field or record:	Corresponds to this RECEIVED parameter:
MSGNAME	MSGNAME
MSGSEQN	MSGSEQNO
MSGSEQO	MSGSEQO
MSGDATE	MSGDATE
MSGTIME	MSGTIME
MSGTZONE	Because this field is always L, Expedite Base/MVS does not provide a parameter for it.
MSGGRPIN	Expedite Base/MVS does not provide this information.

T records (received)

expEDIte/MVS Host places data in T records following an RMH record. Expedite Base/MVS places data after the RECEIVESTREAM command in the message response file (ddname OUTMSG) and ends your data with the string you specified in the ENDSTR parameter.

Depending on the design of your application, you might have to make an adjustment, because RECEIVESTREAM does not start every data record with a T. If you use fixed-length, 80-byte records, RECEIVESTREAM receives one more byte per

T record receive comparison

Examples of response files used to receive T records are:

expEDIte/MVS Host OUTFILE

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CRMDACT1 USER002 TEST1
RGHACT1 USER002 STEST1 1EXMV000T 131
RHI E N JYR8URYD 20001
RFN
RFL
D
RMH 000026910314153909LL
TTHIS IS LINE ONE OF THE DATA
TTHIS IS LINE TWO OF THE DATA
TTHIS IS LINE THREE OF THE DATA
```

Expedite Base/MVS OUTMSG

```
RECEIVESTREAM ACCOUNT(ACT1) USERID(USER002) CLASS(TEST1) ENDSTR(XXX);
RECEIVED ACCOUNT(ACT1) USERID(USER002) CLASS(TEST1) CHARGE(6)
MSGDATE(980314)
MSGDATELONG(19980314) MSGTIME(153909) MSGSEQO(000026)
SESSIONKEY(HXK9H4NW)
SYSNAME(EXMV000T) SYSLEVEL(440) DATATYPE(E)  EDITYPE(UNFORMATTED)
RECFM(????)
RECLEN(0) RECDLM(N) UNIQUEID(JYR8URYD) SYSTYPE(20) SYSVER(4);
THIS IS LINE ONE OF THE DATA
THIS IS LINE TWO OF THE DATA
THIS IS LINE THREE OF THE DATA
XXX
```

Defining lists

With expEDIte/MVS Host, you define a distribution list by placing list define records in a list data set. The name of the list is the ddname that you use to reference the list data set. With Expedite Base/MVS, you place a LIST command in the message command file (ddname INMSG).

List record

An expEDIte/MVS Host list record contains a maximum of four user IDs. If your list has more than four members, you must use multiple list records. Use a single Expedite Base/MVS LIST command to define a list. You must specify an ALIAS/ALIASNAME or a SYSID/ACCOUNT/USERID parameter combination for every member of the list. Specify the name of the list with the LISTNAME parameter.

List data sets

With Expedite Base/MVS, lists must be defined in INMSG. However, you may keep your LIST commands in separate data sets, if necessary. Do this by concatenating multiple data sets for the INMSG ddname. An example is provided here.

```
//INMSG  DD *,DCB=USER01.LIST1
//       DD DSN=USER01.LIST1,DISP=SHR
//       DD DSN=USER01.MYLIST,DISP=SHR
//       DD *
SEND LISTNAME(MYLIST) FILEID(USER01.ORDER.DATA) CLASS(TEST1);
```

The list named MYLIST must be defined in a LIST command contained in either USER01.LIST1 or USER01.MYLIST.

The first statement in the example is necessary to allow standard data sets to be concatenated before a SYSIN data set. You must use a DD statement like the one previously illustrated if you intend to use a SYSIN data set for your INMSG. Also, each of the concatenated data sets must have the same record length and record format. The first data set in the concatenation must have a block size at least as large as subsequent data sets.

List comparison

Examples of command files used to define lists are provided here.

expEDIte/MVS Host MYLIST

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
ACT1 USER002 ACT1 USER003 ACT1 USER004 ACT1 USER005
ACT1 USER006 ACT1 USER007 ACT1 USER008 ACT1 USER009
```

Expedite Base/MVS INMSG

```
LIST LISTNAME(MYLIST)
ACCOUNT(ACT1) USERID(USER002) ACCOUNT(ACT1) USERID(USER003)
ACCOUNT(ACT1) USERID(USER004) ACCOUNT(ACT1) USERID(USER005)
ACCOUNT(ACT1) USERID(USER006) ACCOUNT(ACT1) USERID(USER007)
ACCOUNT(ACT1) USERID(USER008) ACCOUNT(ACT1) USERID(USER009);
```

Retrieving audits

With expEDIte/MVS Host, you use the CAR command to place audit information in your mailbox. expEDIte/MVS Host gives you an RAR record in response to a CAR command. With Expedite Base/MVS, you use the AUDIT command, and Expedite Base/MVS gives you a RETURN record.

CAR command

The following table shows which Expedite Base/MVS AUDIT command parameters correspond to the old CAR fields.

This CAR field:	Corresponds to this AUDIT parameter:
RECTYPES	MSGTYPE
DESTACCT/DESTUID/ DTBLTYP/DTBLID	All of these fields are used to specify a user ID. Use the ALIAS/ALIASNAME or SYSID/ACCOUNT/USERID parameter combinations to specify the user ID.
DATEFROM	STARTDATE
DATETO	ENDDATE
MSGUCLS	CLASS
TIMEZONE	TIMEZONE
STATUS	STATUS
ALTUSER	ALTUSERID

Audit comparison

Examples of command files used to retrieve audits are:

expEDIte/MVS Host

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CARB ACT2 USER005
```

Expedite Base/MVS

```
AUDIT ACCOUNT(ACT2) USERID(USER005);
```

Canceling messages

With expEDIte/MVS Host, you use the CCN command to cancel messages before delivery. With Expedite Base/MVS, you use the CANCEL command.

CCN command

The following table shows which Expedite Base/MVS CANCEL command parameters correspond to the old CCN fields.

This CCN command field:	Corresponds to this CANCEL parameter:
DESTTYP/DESTACCT/ DESTUID/DTBLTYP/ DTBLID	Use the SYSID/ACCOUNT/USERID, ALIAS/ ALIASNAME, or LISTNAME parameters as needed to specify this information.
MSGNAME	MSGNAME
MSGSEQN	MSGSEQNO
MSGUCLS	CLASS

MSGDCLS	PRIORITY
SUBDATE	STARTDATE
SUBTIME	STARTTIME
ENDDATE	ENDDATE
ENDTIME	ENDTIME
TIMEZONE	TIMEZONE
MSGRCPT	ACK

Cancel comparison

Examples of command files used to cancel messages before delivery are:

expEDIte/MVS Host

```
----+----1----+----2----+----3----+----4----+----5---+----6----+----7----+----
CCNDACT3 USER004 TESTCLS
```

Expedite Base/MVS

```
CANCEL ACCOUNT(ACT3) USERID(USER004) CLASS(TESTCLS);
```

PASSTHROUGH command

With expEDIte/MVS Host, you can use the PASS-THROUGH command to request Information Exchange services that are not supported with actual expEDIte/MVS Host commands. With Expedite Base/MVS, these services are provided with new commands. The following table shows the PASS-THROUGH functions and the Expedite Base/MVS commands that correspond to them.

This PASS-THROUGH function:	Is performed by this Expedite Base/MVS command:
Archive retrieve	ARCHIVEMOVE
List define	LIST
List verify	LISTVERIFY
Load test messages	TESTMSG
Message inquiry	MSGINFO
Session inquiry	SESSIONINFO
Define alias	DEFINEALIAS

ENQUEUE command

With expEDIte/MVS Host, you can use the ENQUEUE command to reserve a list of Information Exchange user IDs for use in an expEDIte/MVS Host session.

Expedite Base/MVS automatically enqueues on the user ID before it starts an Information Exchange session, and automatically dequeues when it ends the session. Therefore, no ENQUEUE command is needed.

NOTE: The enqueue is only in effect for the MVS system where Expedite Base/MVS is running.

Comparison summary

This section compares a complete expEDIte/MVS Host job with a complete Expedite Base/MVS job. Both jobs perform the same tasks.

expEDIte/MVS Host job

```
//jobname JOB (insert your installation-specific job statement)
//JOBLIB DD DSN=expedite.mvs.loadlib,DISP=SHR
//TP EXEC PGM=TPMAIN,REGION=1024K,
// PARM='NOSPIE,NOSTAE/myapplid,ieapplid,,,,,,Y'
//INFILE DD *
CSSACT1 USER01 USER01 W0000 RESETSES
CSFLMYLIST          3        CTESTCLAS
CSFAUSER01.TEST.DATA
CSP 3   EDITEST
CSPAUSER01.EDI.TEST
CGHDACT1 USER002 TESTCLAS 3
CMHTESTNAME
TLINE ONE OF THE DATA
TLINE TWO OF THE DATA
CRFDACT1 USER002 TESTCLAS
CRFC AUSER01.TEST.OUT
CRPDACT1 USER002 EDITEST
CRPAUSER01.EDI.OUT
CRMDACT1 USER002 CLASS1
CCNDACT3 USER004 TESTCLS
CARB ACT2 USER005 L
CSE
//QUTTABLE DD *
X 01 GX01 TTABLE01
E 02 GE02 TTABLE02
/*
//TTABLE01 DD *
DUNS01 ACT2 USER001
DUNS02 ACT3 USER005
/*
//OUTFILE DD SYSOUT=*
//MYLIST  DD DSN=USER01.MYLIST,DISP=SHR
//ISCMSGS  DD SYSOUT=*
//PLIDUMP  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//ISCTRACE DD DSN=USER01.TRACE,DISP=OLD
//SYSPRINT DD SYSOUT=*
```

Expedite Base/MVS job

```
//jobname JOB (insert your installation specific job statement)
//JOBLIB  DD DSN=expedite.mvs.loadlib,DISP=SHR
```

```
//TP      EXEC PGM=IEBASE,REGION=0K
//INMSG   DD *

START ACCOUNT(ACT1) USERID(USER01) IEPASSWORD(USER01);

LIST LISTNAME(MYLIST)ACCOUNT(ACT1) USERID(USER002) ACCOUNT(ACT1)
   USERID(USER003);

SEND FILEID(USER01.TEST.DATA) LISTNAME(MYLIST) CLASS(TESTCLAS);

SENDEDI FILEID(USER01.EDI.TEST) CLASS(EDITEST);

SENDSTREAM ACCOUNT(ACT1) USERID(USER002) MSGNAME(TESTNAME)
CLASS(TESTCLAS) ENDSTR(XXX);
LINE ONE OF DATA
LINE TWO OF DATA
XXX

RECEIVE FILEID(USER01.TEST.OUT) ACCOUNT(ACT1) USERID(USER002)
   CLASS(TESTCLAS);

RECEIVEEDI FILEID(USER01.EDI.OUT) ACCOUNT(ACT1) USERID(USER002)
   CLASS(EDITEST);

RECEIVESTREAM ACCOUNT(ACT1) USERID(USER002) ENDSTR(XXXX) CLASS(CLASS1);

CANCEL ACCOUNT(ACT3) USERID(USER005) CLASS(TESTCLS);

AUDIT ACCOUNT(ACT2) USERID(USER005);

END;
/*
//INPRO    DD *
IDENTIFY IEACCOUNT(ACT1) IEUSERID(USER01) IEPASSWORD(USER01)
TIMEZONE(W0000);
SNACOMM IELUNAME(IEAPPLID) USERLUNAME(MYAPPLID);
TRANSMIT AUTOSTART(N) AUTOEND(N);
TRACE LINK(Y) PROTOCOL(Y);
/*
//QUALTBL  DD *
DATATYPE(X) QUALIFIER(01) ALIAS(GX01) TTABLE(DD:TTABLE01)
DATATYPE(E) QUALIFIER(02) ALIAS(GE02) TTABLE(DD:TTABLE02);
//TTABLE01 DD *
EDIDEST(DUNS01) ACCOUNT(ACT2) USERID(USER001);
EDIDEST(DUNS02) ACCOUNT(ACT3) USERID(USER005);
/*
//OUTMSG   DD SYSOUT=*
//OUTPRO   DD SYSOUT=*
//BASETRC  DD DSN=USER01.OUTTRACE,DISP=OLD
//LINKTRC  DD DSN=USER01.LNKTRACE,DISP=OLD
//ERRORMSG DD DSN=USER01.ERRORMSG,DISP=SHR
//ERRORTXT DD DSN=USER01.ERRORTXT,DISP=SHR
//SYSPRINT DD SYSOUT=*
```

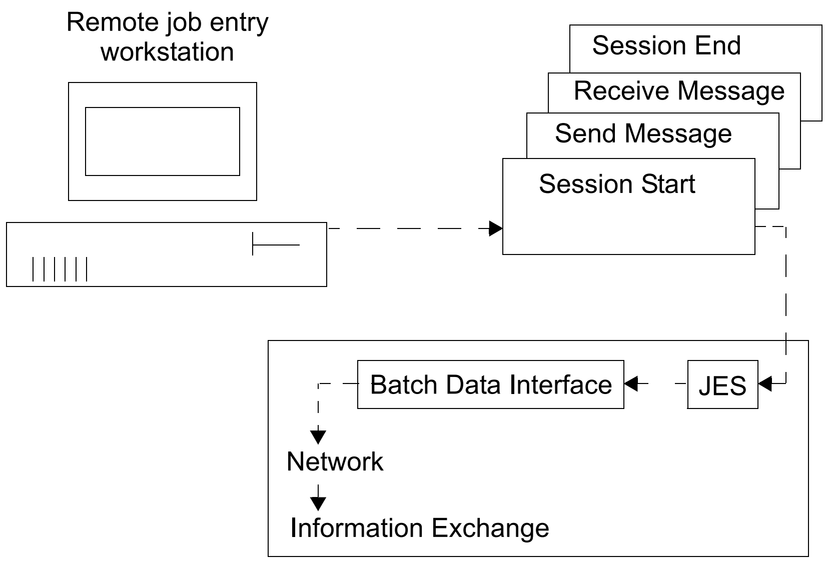
Batch data interface

Expedite Base/MVS is installed on an network MVS system. You can send and receive Information Exchange messages by submitting an Expedite Base/MVS job from a remote job entry (RJE) terminal. When Expedite Base/MVS is used in this way, it is called the Batch Data Interface (BDI).

BDI is a copy of Expedite Base/MVS that runs on the network as a service for remote job entry users. RJE workstations communicate with Information Exchange through the Job Entry Subsystem (JES) and the Information Exchange BDI.

The Information Exchange BDI is a file-oriented program that runs as a batch job and provides all of the necessary protocols. Your system only needs to provide the commands, which are run in the order in which they occur.

The following figure shows the flow of a BDI job from an RJE workstation through the BDI to Information Exchange.



Because BDI runs on a network processor, you must perform the following tasks before using BDI:

- Be sure that your RJE terminal can communicate with the network.
- Get a user ID and password on the network MVS system.
- Get the names of the data sets that contain the IEBASE program and the ERRORTXT, ERRORMSG, and INPRO files.

Your marketing representative can help you get the necessary information.

Information Exchange BDI commands are free-format, but are entered as 80-byte, fixed-length records. Message text can be read as SYSIN (inline) data or from a sequential data set or a partitioned data set member stored in the MVS processor. Output is in 80-byte, fixed-length records.

Modifying the Batch Data Interface job control language

This section describes the steps for modifying the Batch Data Interface (BDI) job control language and provides an example of a BDI job.

An example of an Information Exchange BDI job follows:

```
//jjjjjjjj JOB ,'uuuuuuu',NOTIFY=uuuuuuu,USER=uuuuuuuu,
// PASSWORD=pppppppp
/*ROUTE PRINT your rje terminal
//EXP EXEC PGM=IEBASE,REGION=0K
//INMSG DD *
#Identify message commands here.
#This example is receiving all files into the given DD.
receive fileid(DD:RCVFILE);
//*Note: the concatenated INPRO dataset, this contains the SNACOMM
//*command needed to provide the userlu names.
//INPRO DD DISP=SHR,DSN=SYS2.EBMVSPRD.BDILU
```

```
// DD *
#Provide the IE account, userid, and password to be used for
#the IE session start
identify ieaccount(act1) ieuserid(user01) iepassword(paswrd)
timezone(est);
//* THE FOLLOWING JCL STATEMENTS ARE WHERE YOU SPECIFY
//* THE OUTPUT AREAS TO BE USED
//OUTMSG DD SYSOUT=*
//OUTPRO DD SYSOUT=*
//RCVFILE DD SYSOUT=B,DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//SYSPRINT DD SYSOUT=*

//* The error messages and error text data sets residing on the
//* Information Network system
//ERRORMSG DD DISP=SHR,DSN=SYS2.EBMVSPRD.ERRORMSG
//ERRORTXT  DD DISP=SHR,DSN=SYS2.EBMVSPRD.ERRORTXT
```

The following steps describe the modifications necessary for the Information Exchange BDI JCL.

1. Complete the JOB statement.

   The JOB statement shown in the previous JCL example requires these substitutions:

jjjjjjj	The name of the job to be run. This name should be the MVS user ID followed by a single byte.
USER=uuuuuuu	The MVS user ID.
PASSWORD=ppppppp	The MVS password valid for the user ID.

   NOTE: In order to change your MVS password from your Information Exchange BDI JCL, the format of the password parameter should be:

   ```
   PASSWORD=(oldpswd,newpswd)
   ```

   This version of the password parameter should only be used to change to a new password. Use the standard password parameter at all other times.

   Additional information on JOB statement requirements may be found in an MVS JCL reference.

2. Ensure that JOBLIB and STEPLIB are not used.

   Do not code a JOBLIB or STEPLIB DD in the Information Exchange BDI JCL. The IEBASE program is included in the Information Exchange MVS LINKLIST.

3. Specify the INPRO area.

   Normally, your system programmer would provide you with the logical unit (LU) names for the SNACOMM command. However, when you use the Information Exchange BDI, you must concatenate a data set that resides on the MVS system to your INPRO, for the LU names that are defined to VTAM on the Information Exchange MVS system. The data set name is SYS2.EBMVSPRD.BDILU. This data set contains a SNACOMM command that names a list of defined LUs, from which Expedite Base/MVS will select the first unused LU name. For more information, see “SNACOMM command” on page 71.

   This data set is concatenated with the user’s own INPRO DD, containing the IDENTIFY command and optional TRACE and TRANSMIT commands. The user’s own INPRO file can be found inline, as illustrated in the next JCL example, or in an MVS data sets. For more information, see “MVS disk data sets” on page 343.

4. Specify ERRORTXT and ERRORMSG files.

   Because Expedite Base/MVS runs on an Information Exchange processor for Information Exchange BDI customers, the ERRORTXT and ERRORMSG data sets also reside on the Information Exchange processor. These data set names need to be included in the JCL for Information Exchange BDI.

   - ERRORTXT SYS2.EBMVSPRD.ERRORTXT
   - ERRORMSG SYS2.EBMVSPRD.ERRORMSG

5. Specify an input area.

   The statements beginning with //INMSG and //INPRO in the following example are the statements in which you specify an input area. You can specify an MVS data set name, or the Information Exchange BDI commands can be specified (inline).

   The following is an example of the JCL using an MVS data set name for entering the BDI commands:

```
//USER1X JOB ,USER=USER1,PASSWORD=USER1PWD
//EXP  EXEC PGM=IEBASE,REGION=0K
//INMSGDD DISP=SHR,DSN=USER1.DATA.SET.NAME(MEMBER1)
//INPRO DD DISP=SHR,DSN=SYS2.EBMVSPRD.BDILU
//     DD DISP=SHR,DSN=USER1.INPRO.DATASET.NAME
         *
         *
         *
```

   The following is an example of the JCL required to use inline commands:

```
//USER1X JOB ,USER=USER1,PASSWORD=USER1PWD
//EXP  EXEC PGM=IEBASE,REGION=0K
//INMSG DD *
#Receive data from account act2, userid user004 with
#user class orders
#End the data with the string xxxx
receivestream account(act2) userid(user004) endstr(xxxx)
   class(orders);
//INPRO DD DISP=SHR,DSN=SYS2.EBMVSPRD.BDILU
//     DD *

identify ieaccount(act1) ieuserid(user01) iepassword(paswrd)
```

6. Specify an output area.

   The statements beginning with //OUTMSG, //OUTPRO, //RCVFILE, //LINKTRC, and //BASETRC in the following JCL example all specify output areas. You can specify an MVS data set, print the BDI output, or use any other valid MVS output method. For more information, see “MVS disk data sets” on page 343.

   The following is an example of the JCL using an MVS data set name for writing the BDI responses:

```
//USER1X JOB ,USER=USER1,PASSWORD=USER1PWD
//EXP  EXEC PGM=IEBASE,REGION=0K
//INMSG DD DISP=SHR,DSN=USER1.BDI.INPUT(MEMBER1)
```

```
//OUTMSG DD DISP=SHR,DSN=USER1.BDI.OUTMSG(MEMBER2)
//OUTPRO DD DISP=SHR,DSN=USER1.BDI.OUTPRO
//RCVFILE DD DISP=SHR,DSN=USER1.BDI.RCVFILE(MBRNAME)
//LINKTRC DD DISP=SHR,DSN=USER1.BDI.LINKTRC
//BASETRC DD DISP=SHR,DSN=USER1.BDI.BASETRC
         *
         *
         *
```

The following is an example of the JCL required to route the output to an RJE punch file:

```
//ddname DD SYSOUT=B
```

The following is an example of the JCL required to route the output to a printer:

```
//ddname DD SYSOUT=A
```

The following is an example of the JCL required to route the output listing:

```
//ddname DD SYSOUT=*
```

MVS disk data sets

Since any MVS data set used in the Information Exchange BDI job must reside on the MVS system, the customer must purchase disk space to specify an MVS data set name on either the input or output areas.

Your Global Services sales representative can help with ordering the required disk space.

Checkpoint-level, file-level, and user-initiated recovery with BDI

Since checkpoint-level, file-level, and user-initiated recovery require permanent files, Information Exchange BDI users must purchase disk space on the network to use these recovery methods. For more information, see Chapter 5, “Using checkpoint-level, file-level, and user-initiated recovery.” Session-level recovery is available to all Information Exchange BDI users.

Problem determination JCL

If the Customer Care Help Desk support is needed for problem determination, the following JCL can be used. The user can add MSGCLASS=X and TYPRUN=HOLD to the JCL in the area specified in the example. If one or both of these parameters are specified, Customer Care Help Desk personnel will be able to inspect the job input and output.

Input JCL

```
//jobname JOB ,'userid',NOTIFY=userid,USER=userid,
// PASSWORD=mvspswd,TYPRUN=HOLD,MSGCLASS=X
//*******************************************************************//
//* PURPOSE  Parameters added for problem //
//* determination //
//*VARIABLES //
//* jobname Any 8 character or less name //
//* userid  Submitter's MVSPS userid //
//* mvspswd Submitter's MVSPS password //
//* acctid  ACCOUNT of user whose error msgs //
//* are to be received //
//* userid  USERID of user whose error msgs //
//* are to be received //
//* password  IE PASSWORD of user whose error //
//* messages are to be received //
//******************************************************************//
//EXP     EXEC PGM=IEBASE,REGION=0K
//INMSG   DD *
#Send the file referenced by DD name DD1 to alias name
#jones in alias table ptb1 with user class notes
send fileid(dd:dd1) alias(ptb1) aliasname(jones) class(notes);
//INPRO   DD DISP=SHR,DSN=SYS2.EBMVSPRD.BDILU
//        DD *
identify ieaccount(acctid) ieuserid(ieid) iepassword(password)
timezone(est);
/*
//OUTMSG  DD SYSOUT=*
//OUTPRO  DD SYSOUT=*
//DD1     DD *
Here is the data to be sent with the send file command.
Here is line two of the data.
/*
//ERRORMSG   DD DISP=SHR,DSN=SYS2.EBMVSPRD.ERRORMSG
//ERRORTXT   DD DISP=SHR,DSN=SYS2.EBMVSPRD.ERRORTXT
//SYSPRINT   DD SYSOUT=*
```

RJE device and host setup definitions

Most users use their RJE device to communicate with a spool control application, such as JES2 or POWER. Bisynchronous (BSC) dial access does not impose any restrictions on which applications are supported, so any application that supports an IBM 3776 as an LU type 1 can communicate with a device using BSC dial access. If you are not sure about the communications setup at your site, call Customer Care to schedule an attachment test.

RJE devices and bisynchronous (BSC) dial access

The RJE devices all use the same physical port when using BSC dial access. Because of the differences in the supported RJE devices (2770, 2780, and 3780), the calling RJE devices must specify which device type is using BSC dial access. All RJE devices supported using BSC dial access have the option to send a terminal ID after establishing the dial connection. On emulating devices, it is usually specified in the emulator definition. The default device is the 3780. For more information, refer to *Using RJE on the Information Services Information Network*.

The ID protocol sequence (ID ENQ) is treated as a normal bid from the terminal. Any additional data sent during this time would normally be discarded. To allow the terminal user to modify certain port characteristics, the sequence “STX RJEx ETX” is allowed before the bid (ENQ) is terminated (EOT). In the RJEx data field, x can be in the range 1 to 9, with the following optional support based on terminal type:

- 2770
  A standard I/O block size of 256 is used for this device unless option 1/2 is specified in the RJEx option data. Option 1 (RJE1) selects a block size of 128 bytes, and option 2 (RJE2) selects a block size of 512 bytes.

- 2780
  A standard I/O block size of 256 is used for this device unless option 1/2 is specified in the RJEx option data. Option 1 (RJE1) selects a block size of 200 bytes, and option 2 (RJE2)

selects a block size of 400 bytes.

- 3780
  A standard I/O block size of 256 is used for this device unless option 1/2 is specified in the RJEx option data. Option 1 (RJE1) selects a block size of 128 bytes, and option 2 (RJE2) selects a block size of 512 bytes.

Host definitions of RJE devices using BSC dial access

All bisynchronous RJE devices using BSC dial access appear to the network and your processor as an IBM 3776 LU 1 SNA RJE station.

The following restrictions apply to the terminals (2770, 2780, 3780) supported using BSC dial access:

- No Multileaving
  Concurrent transmit and receive is not supported between the application and the RJE station.
- No console or terminal interrupt capability (attention function)
  The operator cannot signal the host application to stop transmitting when the terminal attempts to transmit. The attention function is usually associated with an operator console.
- No compaction/decompaction
  The protocol converter rejects any bind from an application that specifies compaction.
- I/O devices limited to reader/punch, and printer
  Most users will have their RJE devices communicate bisynchronously. The protocol converter will translate header data that is required to direct output and input to the reader, punch, or printer. The RJE terminal or the RJE host application requires this information to determine whether a punch, reader, or print operation is being processed. The format (BSC or SNA) will be based on the direction of the data.
- Blank and data compression/decompression supported
  The network does not compress or decompress data when transmitting to the host application. However, the data received from the host application is compressed or decompressed as follows:
  - IBM 2770 (decompression of all data including blanks)
  - IBM 2780 (decompression of all data including blanks)
  - IBM 3780 (decompression of all data excluding blanks)

Batch Logon Interface

The Batch Logon Interface allows programmable devices to interact with the Session Control application in a consistent manner.

The functions provided by this interface are:

- Logon processing at the Session Control application level, which includes account and user ID verification, password verification, and password maintenance.
- Verification of the process selected (ensures that the selected application is on the user's profile).
- Isolation of the application LU name from the user.

- Ability of the product owner to control use of the product by use of product access and session limit.

To use the Batch Logon Interface, the user must have the following:

- Account name
- Network user ID and password
- Product segment on the network user ID

  The product segment must have the appropriate LU name of the application.

Batch logon request format

The format of the batch logon request is:

```
/*LOGON account,userid,pswd/new pswd/verify new pswd
/*SELECT product/*USERDATA RMTxxx,,rmtpw
```

NOTE: The request should consist of one record. It is on two lines here because of space limitations.

account	The account that the user is registered to use.
userid	The user ID that the user is registered to use.
pswd	The current password for the Session Control application.
new pswd	A request to change the password for the Session Control application.
verify new pswd	Verification of the new password to ensure that an error was not made entering the new password. This is an optional parameter.
product	The application synonym from the user's profile to which this terminal is to be passed.
userdata	RMT*xx* is the remote number and rmtpw is the remove password.

Logging on to the Session Control application

After the session with the Session Control application is started, the welcome message is sent and terminated with an XON character. At this time, the user should send the entire Batch Logon request. If the Logon is unsuccessful, an error message from the Session Control application returns, preceded by SVM00xxx. There is no confirmation of a successful logon to the Session Control application.

Processing the logon request involves two steps. The first step is to parse the request. Any error resulting from this process is sent to the terminal and followed immediately by a line disconnect. The parser does not scan beyond the first error.

After parsing is complete, the next step is to log on to the Session Control application and request an application. Errors that can occur in this step are logon failures and errors in attempting to access the application. These errors result in an error message and a line disconnect.

The Session Control application welcome message has the following format:

```
X'150D'
X'150D'X'150D'WELCOME TO THE INFORMATION NETWORK
X'150D'TERMID: XXXXXXXX YYMMDD HHMMSS
X'150D'CUSTOMER ASSISTANCE: xxxxxxxxxxxxx
X'150D'
X'150D'ENTER "HELP" FOR ASSISTANCE.
X'150D'
X'150D'ENTER USERID ACCOUNT
X'150D'==>
```

The X'150D' is a new line control character followed by a carriage-return character. This causes the cursor to move one line down and return to the left of the screen. The X'150D' is not displayed on the device, but is shown above for users who may need to write a program interface.

Password maintenance

The Session Control application uses RACF to maintain and verify passwords. The restrictions on password selection are:

- The password must be 5 to 8 alphanumeric characters.
- The password must begin with an alphabetic character.
- The password must contain at least three different characters.
- The password must differ from the previous three passwords.

A password can be changed every time the user logs on to the system.

Syntax error messages

SVM00811 Invalid request.

User Response: The first characters of the request were not /*LOGON or /*LOGOFF. Resubmit the request with the first characters as either /*LOGON or /*LOGOFF.

SVM00821 Account number missing or invalid.

Explanation: You did not specify an account name after the LOGON parameter, or the account name you specified was invalid. Account names must be between 1 and 8 characters in length, and must be placed after the LOGON parameter and before the first comma.

User Response: Specify a valid account name and resubmit the request.

SVM00823 User ID missing or invalid.

Explanation: You did not specify a user ID after the ACCOUNT parameter, or the user ID you specified was invalid. User IDs must be between 1 and 7 characters in length, and must be placed after the ACCOUNT parameter and before the second comma.

User Response: Specify a valid user ID and resubmit the request.

SVM00824 Password missing or invalid.

Explanation: You did not specify a password after the USERID parameter, or the password you specified was invalid. Passwords must be between 1 and 8 characters in length, and must be placed after the USERID parameter and before the second slash (/).

User Response: Specify a valid password and resubmit the request.

SVM00825 New password invalid.

Explanation: The length of data between the password and the next slash was greater than 8 characters. Passwords must be between 1 and 8 characters in length.

User Response: Specify a valid password and resubmit the request.

SVM00826 Product missing or invalid.

Explanation: The Service Manager did not find the /*SELECT keyword after the password or new password parameter, or /*SELECT was found, but the product name parameter did not contain between 1 and 8 characters.

User Response: Specify a valid product name and resubmit the request.

SVM00827 Verify password invalid.

Explanation: The length of data in the verify new password is greater than 8 characters, and the Service Manager could not verify your new password. Passwords must be between 1 and 8 characters in length.

User Response: Ensure that the verify new password does not contain more than 8 characters and resubmit the request.

SVM00828 New/verify passwords not equal.

Explanation: The value you specified in the new password parameter does not equal the data in the verify password parameter.

User Response: Ensure that the values you specify in the new password and verify new password parameters are the same and resubmit the request.

Logon processing errors

SVM00831 Profile not found.

Explanation: The network profile for the account and user ID on your Batch Logon request could not be located in the Service Manager profile database.

User Response: Contact your marketing representative to obtain a valid account and user ID.

SVM00836 Product not on your profile.

Explanation: The product you requested on the Batch Logon request is not in the profile for your account and user ID.

User Response: Specify a new product and resubmit the request, or specify ???? to obtain access to the product for your account and user ID.

SVM00841 User ID already logged on.

Explanation: Your user ID is logged on to the AT&T Global Network.

User Response: None.

SVM00851 User ID not defined to RACF.

Explanation: The user ID entered in the batch logon request is not enrolled to RACF.

User Response: Contact the Customer Care Help Desk to determine why the user ID is not defined to the RACF database.

SVM00852 Password not authorized.

Explanation: RACF rejected your logon attempt, because the password you specified on the Batch Logon request is not the correct password for the account and user ID entered.

User Response: Specify a valid password for the account and user ID specified and resubmit the request.

SVM00853 Password expired.

Explanation: RACF rejected your logon attempt, because the password you specified on the Batch Logon request is not the correct password for the account and user ID entered.

User Response: Specify the correct password and resubmit the request, or contact your marketing representative.

SVM00854 New password invalid.

Explanation: A parameter is in the wrong position and the system could not parse the command.

User Response: Place the new password after the current one and separate the passwords with a slash (/). Passwords must be between 1 and 8 characters in length.

SVM00855 User access revoked.

Explanation: RACF rejected your logon attempt. The user ID you specified on the Batch Logon request has been revoked because you attempted to logon to the system three times with an invalid password.

User Response: Contact the Customer Care Help Desk.

SVM00856 Password must be extended.

Explanation: The user is defined to the Service Manager as a user requiring an extended password, but the password supplied contains no extended characters. Valid extended characters are . ? ; ' " ( ) : &.

User Response: Verify the password contains at least a single extended character.

SVM00857 New password must be extended.

Explanation: The user is defined to the Service Manager as a user requiring an extended password, but the password supplied contains no extended characters. Valid extended characters are . ? ; ' " ( ) : &.

User Response: Re-enter the extended password and retry your request.

SVM00859 Password contains an invalid character.

Explanation: The password entered contains special character, which is invalid to RACF.

User Response: Verify the extended character is one of the following legal RACF characters: . ? ; ' " ( ) : &.

SVM00860 New password contains an invalid character.

Explanation: The password entered contains a character which is invalid to RACF.

User Response: Verify the characters in the password are all legal RACF password characters.

SVM00862 Product access denied.

Explanation: RACF rejected your logon attempt, because the owner of the application you requested has revoked the access for the account or user ID on the Batch Logon request.

User Response: Contact the Customer Care Help Desk.

SVM00863 Product not available.

Explanation: RACF rejected your logon attempt, because the product you requested is unavailable.

User Response: Contact the Customer Care Help Desk to determine when the product you requested will be available, or specify another product and resubmit the request.

SVM00864 Account/user ID invalid for this terminal.

Explanation: The user has attempted to access the network via a port/terminal to which he is not authorized. Either the user is attempting to use a port/terminal to which he should not be authorized or the RACF database needs to be updated to allow the user access to the port/terminal.

User Response: Contact your Service Administrator to determine which terminal you are allowed to use.

SVM00865 New password is invalid.

Explanation: RACF rejected the logon attempt, because the Service Manager recognizes the new password in the Batch Logon request as a reserved word. A network password must have the following characteristics:

- It must be 5 to 8 characters in length.
- It must contain three different characters.
- It must not be the same as any of your three previous passwords.
- It must begin with an alphabetic character.
- It must not be the word “cancel.”

Extended passwords must contain one of the following characters: ; x & ( ) : " ' ?.

User Response: Specify a valid new password and resubmit the request.

SVM00866 Too many users logged on using this ID.

Explanation: This user has attempted to log on with a generic user ID, which currently has logged on to it the maximum number of allowable users.

User Response: Retry the request after one or more users of the generic user ID have logged off the network.

SVM00867 Logon not permitted. User has reached the logon maximum.

Explanation: Defined to the Service Manager is a parameter specifying the maximum number of times a user can be logged on to a single Service Manager with the same user ID. This limit has been exceeded (for a U.S. language ID).

User Response: The user must wait until one of the existing sessions is terminated before network access will be granted.

SVM00868 No access to product from this type of terminal.

Explanation: The product is full screen, not display mode, and cannot be accessed from a line mode or LU 6.2 device.

User Response: Verify the name of the product being accessed. If it is correct, call the Customer Care Help Desk to determine why the application is defined as not allowing line mode access. Otherwise, switch to a full screen device and retry your request.

SVM00870 Incomplete definition for user ID uuuuuuuu.

Explanation: There is no RACF definition for this user ID.

User Response: Contact your Service Administrator.

SVM00871 Logon error occurred for user ID uuuuuuuu.

Explanation: An unexpected RACF error occurred during the logon process.

User Response: Contact the Customer Care Help Desk so they may determine the cause of the failure, which is noted in the Service Manager log (i.e., the user ID may not be defined to RACF).

Glossary

A

abend. Abnormal end of task, as in “the session abends.”

account name. A name that identifies an account to a program, device, or system.

acknowledgment. A response from Information Exchange that lets you know whether your messages were delivered, received, purged, or various combinations of these three.

address. (1) A unique code assigned to a user connected to a network. (2) The location in the storage of a computer where data is stored.

advanced interactive executive (AIX). An operating system that serves as an interface between users and Information Exchange.

AIX. Advanced Interactive Executive.

alias. An alternate name used in place of an account and user ID.

alias table. An alternate name file. Expedite Base/MVS uses alias tables to resolve EDI destinations.

alphanumeric. Generally, any keyboard character, but for practical purposes should be restricted to alphabetic, numeric, space, and common punctuation characters.

American National Standard Code for Information Interchange (ASCII). The standard code, using a coded character set consisting of 7-bit coded characters (8 bits include parity check), used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.(A)

application program interface (API). The formally defined programming language interface used by an IBM system control program or a licensed program to communicate with the user of the program.

archive. A place to store messages on a database for future reference.

ASCII. American National Standard Code for Information Interchange.

asynchronous. A protocol that permits a communication device to operate in an unsynchronized and unpredictable manner, much like a human conversation; used for modems and low-speed ASCII terminals (PCs).

attribute. A property or characteristic of one or more entities, for example, length, value, color, or intensity.

audit trail. A way of tracking and verifying basic information about the status of messages.

B

base trace file (BASETRC). The file that provides Expedite Base/MVS trace information other than LINK trace.

BASETRC. Base trace file.

Batch Data Interface (BDI). A product on the network that allows users to communicate with Information Exchange if they have a remote job entry processor or terminal.

BDI. Batch Data Interface.

binary file. A file, such as an executable computer program, that contains machine instructions that a person cannot read and that cannot be entered from a computer keyboard.

binary synchronous (bisynchronous). A protocol that allows a communication device to operate in a synchronized and predictable manner; used for high-speed computer-to-computer communication and for mainframe computer-to-nonprogrammable terminal communication.

carriage-return and line-feed characters (CRLF). A formatting control that moves the printing or display point to the first position of the next line.

CDH. Common data header.

CDRSC. Cross-domain resource.

centralized alias table. Permanent tables that reside within Information Exchange and contain a centralized list of addresses. You can put a listing of your trading partners' addresses in this table instead of maintaining destination tables in multiple locations. A centralized alias table enables Expedite Base/MVS to resolve destinations, because the table contains a list of EDI destinations paired with Information Exchange destinations. Expedite Base/MVS searches this table for an EDI destination and then uses the corresponding Information Exchange destination as the actual address.

certificate. A digital document that binds a public key to the identity of the certificate owner; thereby, enabling the certificate owner to be authenticated. A certificate is issued by a certificate authority (CA).

certificate authority (CA). The legal entity that issues the digital certificates that identify a certificate owner.

character. Generally, any letter, number, punctuation mark, or other symbol that can be entered at a computer keyboard.

checkpoint-level recovery. A method of restart and recovery within Expedite Base/MVS; a point where information about the status of a job can be recovered so that the job step can be restarted later.

CICS. Customer Information Control System.

command. A request to run a particular program or function.

command file. A file containing Expedite Base/MVS commands. There are two Expedite Base/MVS command files, profile command (ddname INPRO) and message command (ddname INMSG). You place commands pertaining to your profile in INPRO and those pertaining to the transfer of files or data in INMSG.

committal. The point at which a message is either delivered, canceled, or purged. When a session fails, all uncommitted messages are lost.

compression. The process of eliminating gaps, empty fields, and redundant data to shorten the length of files.

common data header (CDH). The first message in a group of messages, the common data header provides information about the messages.

CRLF. Carriage-return and line-feed characters.

Customer Information Control System (CICS). A program that enables transactions entered at remote terminals to be processed at the same time.

D

data definition (DD) name. The name of a data definition (DD) statement that corresponds to a data control block that contains the same name.

data definition statement. A job control statement describing a data set associated with a specific job step.

default. An alternative value an application program uses automatically when none has been specified.

delivery acknowledgment. An acknowledgment generated by Information Exchange when a destination user receives a message from the Information Exchange mailbox.

destination. In a network, any point or location, such as a node, station, or a particular terminal, to which information is sent.

distribution list. A list of the addresses of users with whom a certain user communicates. It is used to send messages to several people without having to type their addresses.

E

EBCDIC. Extended binary-coded decimal interchange code.

EDI. Electronic data interchange.

EDI destination table. A list of EDI destinations paired with Information Exchange destinations used by Expedite Base/MVS.

EDIFACT. An electronic data interchange standard developed by the United Nations Economic Commission for Europe.

EDI qualifier table. A list of types of EDI data (X12, UCS, EDIFACT, or UN/TDI) paired with the ID qualifier for a particular type of data (for example, 01 for an X12 DUNS number).

electronic data interchange (EDI). The process of sending specially formatted business documents directly from one computer to another electronically.

electronic data interchange control files. Files used to send EDI data: EDI qualifier tables (QUALTBL) and EDI destination tables (TTABLExx).

envelope. Information Exchange's electronic file that contains the text of a message, together with a message header.

error message file (ERRORMSG). The file that provides Expedite Base/MVS with error descriptions in the RETURN, WARNING, SESSIONEND, and PROFILERC output records.

ERRORMSG. Error message file.

ERRORTXT. Extended error text file.

ESO. Extended Security Option.

extended binary-coded decimal interchange code (EBCDIC). A coded character set consisting of 256 8-bit characters.

extended error text file (ERRORTXT). The file that provides Expedite Base/MVS with extended error descriptions in the RETURN, WARNING, SESSIONEND, and PROFILERC response records.

Extended Security Option (ESO). An option that provides additional password and Information Exchange mailbox security.

F

file-level recovery. A method of restart and recovery within Expedite Base/MVS; checkpoints are taken for each file sent and received.

G

global alias. An alternative name that can be used on a particular system.

global alias table. (1) A system-wide alias table. (2) An alternative name table set up within a system.

H

HFS. Hierarchical file system.

host computer. The primary or controlling computer in a multiple computer installation.

I

ibm0rely. The LU name of the Information Exchange Common Front End in the United States.

Information Exchange base (IEBASE) program. The program that starts Expedite Base/MVS.

Information Exchange. (1) A communication service that permits users to send and receive information electronically. (2) A continuously running CICS application on the network that stores and forwards information to trading partners.

Information Exchange Administration Services. An online, panel-driven product that the Information Exchange Service Administrator uses to perform administrative tasks for Information Exchange.

Information Exchange mailbox. See mailbox.

Information Exchange service administrator. The person who coordinates the use of Information Exchange within a company.

INMSG. Expedite Base/MVS message command file.

INPRO. Expedite Base/MVS profile command file.

J

JCL. Job control language.

job control language (JCL). A control language that identifies a job to an operating system and describes the job’s requirements.

L

leased lines. A connection between systems or devices that does not have to be made by dialing.

library. A place to store information for an extended period of time. A library consists of a collection of files called library members.

link trace file (LINKTRC). The file that provides Expedite Base/MVS with link trace information.

mailbox. A temporary storage area for electronic mail from which data is retrieved by the intended recipient.

member. A file in a library.

message. (1) Any piece of data that users send or receive. (2) The smallest subdivision of information that can be sent from one user to another. (3) An instruction or explanation on the screen that tells you what the system is doing or warns you that the system has detected an error.

message class. A category agreed upon among trading partners that is used to group mail.

message command. Commands that pertain to the transferring of files or data. You place message commands in the message command file (INMSG).

message command file (INMSG). The file that sends and receives files and messages, including EDI data. The ddname of this file is INMSG. You enter file transfer requests with message commands into INMSG.

message group. A collection of messages that is treated as a single entity. A file of records to be printed as a single report is an example of a message group.

message response file (OUTMSG). A file containing the Expedite Base/MVS and Information Exchange replies to certain message commands. The ddname of this file is OUTMSG. Expedite Base/MVS generates this file after it processes the message command file (INMSG). OUTMSG contains return codes, error messages, and completion codes.

O

organizational alias. (1) An alias that can be used by any user in an account. (2) A company-wide alias table.

organizational alias table. An alias table set up within an account.

OUTMSG. The message response file for Expedite Base/MVS.

OUTPRO. The profile response file for Expedite Base/MVS.

P

parameter. (1) A variable that is given a constant value for a specified application and that may denote the application. (2) An item in a menu for which you specify a value or for which the system provides a value when the menu is interpreted. (3) Data passed between programs or procedures.

private alias table. An alias table set up for an individual user.

profile command file (INPRO). A file in which you place profile commands pertaining to profile specific information, such as your passwords, user ID, and account. The ddname for this file is INPRO.

profile response file (OUTPRO). A file containing the Expedite Base/MVS and Information Exchange replies to certain profile commands. The ddname for this file is OUTPRO. Expedite Base/MVS generates this file after processing the profile command file (INPRO). The profile response file contains return codes, error messages, and completion codes.

purge acknowledgment. The acknowledgment generated by Information Exchange when a message reaches the storage time limit and is purged from the receiver's mailbox.

Q

qualifier table. A list of types of EDI data (X12, UCS, EDIFACT, or UN/TDI) paired with the ID qualifier for a particular type of data (for example, 01 for an X12 DUNS number).

R

receipt acknowledgment. The acknowledgment generated by Information Exchange when a message reaches the receiver's mailbox after a successful Expedite Base/MVS session. It appears in the message response file (OUTMSG).

recovery level. The point from which a session needs to be restarted following a failure. Expedite Base/MVS has two recovery levels: session and checkpoint-level recovery.

response file. An output file into which Expedite Base/MVS echoes commands from INPRO and INMSG along with their associated return codes.

S

Secure Socket Layer . A secure method of communicating by way of the Internet.

service administrator. A primary contact person in an organization who controls use of the service by the users within the organization and who assists the service support groups (for example, Customer Care Help Desk).

session. The period of time during which you can communicate with a computer system or one of its programs; usually, the elapsed time between logon and logoff.

session-level recovery. A method of restart and recovery within Expedite Base/MVS; no messages are committed until the session ends normally.

SNA. Systems Network Architecture.

SSL. See Secure Socket Layer.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks. The layered structured of SNA allows the ultimate origins and destinations of information, that is, the users, to be independent of and unaffected by the specific SNA network services and facilities used for Information Exchange.

syntax. The rules for constructing a command.

T

TCP/IP. Transmission Control Protocol/Internet Protocol.

Transmission Control Protocol/Internet Protocol (TCP/IP). A set of communications protocols that support peer-to-peer connectivity functions for both local and wide area networks.

trace file. A file that provides you with a history of transactions. In Expedite Base/MVS there are two types of trace files: base trace (BASETRC) and link trace (LINKTRC).

trace message text file (TRACEMSG). A file containing trace messages in languages other than English.

trading partners. The business associates with whom users of EDI exchange information electronically.

U

UCS. Uniform Communication Standard.

Uniform Communication Standard (UCS). A standard EDI format used in the grocery industry.

United Nations/Trade Data Interchange (UN/TDI). An EDI standard for administration, commerce, and transportation fields developed by the United Nations Economic Commission for Europe.

user-initiated recovery. A method of restart and recovery within Expedite Base/MVS; checkpoints are taken after each COMMIT command, unless there is nothing to commit.

user class. A short description that users can assign to their documents to identify these documents to their trading partners.

user group. A set of users within an account who can communicate with one another.

USERLUNAME. The LU name that Expedite Base/MVS uses when communicating with Information Exchange. Your system programmer must define it to VTAM on your MVS system.

user message class. A category agreed upon among trading partners that is used to group mail.

user profile. A list of the characteristics that describe how a user works with the Information Exchange.

W

wildcard character. A synonym for pattern-matching character.

X

X12. An electronic data interchange standard developed by the American National Standards Institute (ANSI).

Index