z/OS
3.1

TSO/E User's Guide

SA32-0971-60

Note

Before using this information and the product it supports, read the information in “Notices” on page 207.

This edition applies to IBM® z/OS® 3.1 (5655-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2024-01-20

© Copyright International Business Machines Corporation 1988, 2024.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures

Tables

About this document

This book is a general guide for using the TSO/E element of z/OS. It expands the concepts and basic tasks that are presented in the *z/OS TSO/E Primer*.

Who should use this document

Anyone who uses TSO/E should read this book to learn more about TSO/E in relation to command usage and data set management.

All users can refer to *z/OS TSO/E Command Reference*, for detailed reference information. Programmers can learn about the programming aspects of TSO/E in *z/OS TSO/E Programming Guide*, and also *z/OS TSO/E Programming Services*.

How this document is organized

This book is divided into four parts.

- Part 1, "General TSO/E Functions," on page 1 contains information that every TSO/E user must understand, such as logging on and logging off TSO/E, specifying TSO/E commands, naming data sets, and communicating with other TSO/E users.
- Part 2, "Using Data Sets," on page 37 explains how to manage data sets by allocating, freeing, listing, editing, renaming, copying, sending and receiving, printing, and deleting data sets.
- Part 3, "Running a Program," on page 119 explains how to run a previously prepared program and how to execute programs in the foreground and background.

Examples of commands in this book show, in uppercase letters, the invariable parts of the command, such as the command name and operands, and show, in lowercase letters, the variable parts of the command such as data set names and user IDs.

In the following example, TRANSMIT is the command and cannot be changed, DATASET is an operand and cannot be changed, *nodeid.userid* and *test.data* are variables you can change.

```
TRANSMIT nodeid.userid DATASET(test.data)
```

How to use this document

If you have never used this book, read Part 1, "General TSO/E Functions," on page 1 to become familiar with the programming services that TSO/E provides. Then read the chapter that discusses the service you want to use.

Where to find more information

See *z/OS Information Roadmap* for an overview of the documentation associated with z/OS, including the documentation available for z/OS TSO/E.

© Copyright IBM Corp. 1988, 2024

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. For more information, see How to send feedback to IBM.

© Copyright IBM Corp. 1988, 2024

Summary of changes

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line to the left of the change.

Note: IBM z/OS policy for the integration of service information into the z/OS product documentation library is documented on the z/OS Internet Library under IBM z/OS Product Documentation Update Policy (www.ibm.com/docs/en/zos/latest?topic=zos-product-documentation-update-policy).

Summary of changes for z/OS 3.1

This information contains no technical changes for this release.

© Copyright IBM Corp. 1988, 2024

Part 1. General TSO/E Functions

TSO/E [1] is a base element of the z/OS operating system that allows users to interactively work with the system. It is a tool with which you can:

- Communicate with other TSO/E users
- Create an office environment
- Develop and maintain programs in languages such as assembler, COBOL, FORTRAN, PASCAL, PL/I, REXX, and CLIST
- Process data
- Access the MVS operating system

For more information about what TSO/E does, see *z/OS TSO/E General Information*.

This part explains general processes and basic concepts such as:

- Beginning a TSO/E session

  Chapter 1, “Beginning a TSO/E Session,” on page 3 describes logging on to the terminal to begin a TSO/E session and logging off when you want to end the TSO/E session. In addition, it describes types of messages and prompts and ways to respond to them.
- Basic concepts necessary to use TSO/E

  Chapter 2, “Basic Concepts,” on page 11 explains things you need to know to use TSO/E, such as:

  – Commands - TSO/E command syntax and how to issue commands
  – Data sets - types of data sets and how to name them
- Communicating with other TSO/E users

  Chapter 3, “Communicating With Other Users,” on page 25 explains how to send short messages with the SEND command and how to send longer messages with the TRANSMIT command. In addition, it describes how to use nicknames by defining them in a NAMES data set and how to store data in a log.

[1] The name derives from Time Sharing Option/Extensions, because it was an option on previous MVS™ systems to share computing time, and it was functionally extended during its life.

© Copyright IBM Corp. 1988, 2024

Chapter 1. Beginning a TSO/E Session

Your terminal is your link to the computer system that uses TSO/E. Before you start to use TSO/E, you need to understand how to use your terminal.

Beginning a terminal session is relatively simple. You identify yourself to the system and then issue commands to request work from the system. To conduct a TSO/E terminal session you need to:

- Learn about your terminal, including how to enter data, correct mistakes, add or delete characters and lines, and how to interrupt operations from the terminal
- Log on with the TSO/E LOGON command
- Interact with TSO/E
- End the TSO/E terminal session with the TSO/E LOGOFF command

Learning About Your Terminal

All TSO/E terminals have a typewriter-like keyboard. The features of each keyboard vary from terminal to terminal. For example, one terminal may not have a backspace key, while another may not allow for lowercase letters. To learn how to use your terminal, consult the terminal operator's manual that accompanies your device.

Because each installation differs in its access methods, configuration, and procedures, you will have to consult your system programmer or system support center to learn how you can contact your computer system and begin using TSO/E.

Logging On With the LOGON Command

Use the LOGON command to identify yourself to the system and request use of its resources. When you use the LOGON command, the system requires your **TSO/E user ID**, which tells the system who wants to use it.

A user ID can be no longer than seven eight characters and can contain numeric (0-9) and alphabetic (A-Z) characters, but must begin with an alphabetic character. Often a user ID is your name, your initials, your department number, or any combination of valid characters your installation chooses. There are some restrictions on which characters can be used in your user ID, particularly if your system uses JES2. These restrictions are explained in *z/OS TSO/E Administration*.

Depending on your installation, you might also need to specify other information on the LOGON command, such as:

- A **password** that authorizes you to use the system's resources. A password is a group of one to eight characters that you use to prevent other users from using your user ID.
- A **procedure name** that identifies a procedure that initiates your TSO/E session.
- An **account number** that your installation uses to keep track of system usage.
- A **security label** that defines which system resources you can access, and which users you can communicate with. A security label may be 1 to 8 alphabetic or numeric characters or the special characters (#, $, or @). The first character must be alphabetic.

Issuing the LOGON Command

You can issue the LOGON command with no operands. This results in the system prompting you for your user ID.

© Copyright IBM Corp. 1988, 2024

Example

To simply request access to the system, enter:

```
LOGON
```

The system might then prompt you for your user ID with the message:

```
ENTER USERID -
```

You can also issue the LOGON command followed by your user ID. Separate your user ID from the command with a blank character. After you enter this command, the system might prompt you for your password.

Example

If your user ID were YOURID, you could log on as follows:

```
LOGON yourid
```

Your installation determines what appears on your screen after you issue the LOGON command. You might see one of the following things:

- A message or messages, prompting you for information that your installation requires, such as a procedure name, account number, or security label.
- A READY message, indicating that the system recognized your user ID and password. The READY message indicates that you have logged on to the system and can start using TSO/E.

  If users are logged on after using the LOGON command, other things may be displayed instead of the READY message. For example, the logon procedure may automatically invoke ISPF, the Information Center Facility, or an installation-defined panel.
- A full-screen logon panel, on which you can type required information. Your logon panel might look like the following:

```
------------------------------ TSO/E LOGON ----------------------------------
PF1/PF13 ==> Help    PF3/PF15 ==> Logoff    PA1 ==> Attention    PA2 ==> Reshow
You may request specific HELP information by entering a '?' in any entry field.
   ENTER LOGON PARAMETERS BELOW:                  RACF LOGON PARAMETERS:

   USERID    ===> YOURID                          SECLABEL     ===>

   PASSWORD  ===> -                               NEW PASSWORD ===>

   PROCEDURE ===> MYPROC                          GROUP IDENT  ===>

   ACCT NMBR ===> 00123

   SIZE      ===> 5800

   PERFORM   ===>

   COMMAND   ===> EXEC (SETUP)


   ENTER AN 'S' BEFORE EACH OPTION DESIRED BELOW:

          -NOMAIL         -NONOTICE        -RECONNECT         -OIDCARD
```

Note: The SECLABEL field on the full-screen logon panel is displayed only if your installation is using security labels.

When you log on to TSO/E using the full-screen logon panel, the values shown in some of the fields, such as PROCEDURE, ACCT NMBR, and SECLABEL are the values you entered for your previous TSO/E session. You can choose a different value by typing in a new value in the field. The new value you enter is then saved and is displayed whenever you log on again until you either enter a different value or blank out the field.

The values you are required to enter on the logon panel depend on your installation. See your TSO/E administrator for logon instructions.

For more information about logon panels, see Appendix A, "Full-Screen Logon Processing," on page 191.

Interacting With TSO/E

You can use TSO/E in many environments, such as with the Interactive System Productivity Facility/ Program Development Facility (ISPF/PDF), Session Manager, and line mode TSO/E. How you interact with TSO/E depends upon the environment. Line mode TSO/E is the basic TSO/E environment and this book primarily deals with that basic environment.

On most display terminals, when you are in line mode TSO/E, three asterisks, ***, on the screen mean that you are to press the Enter key to continue. You see *** when:

- The screen is full and you need to go to another screen
- A process has completed and the system is ready for your input
- You were temporarily interrupted by a broadcast message and can now resume your work

The READY mode message indicates that the system is ready for a command. When a command executes successfully, you see another READY mode message. When a command executes unsuccessfully, you generally see a message.

Messages

You can receive five types of messages at your terminal:

- Mode messages
- Prompting messages
- Informational messages
- Broadcast messages
- Messages from other users

Note: You may also receive write-to-programmer messages on your screen. These are messages written by the system intended for your system programmer, and usually do not require any action on your part. Chapter 17, "Customizing Your Terminal Session," on page 155 contains information about write-to-programmer messages.

The following topics describe mode, prompting, informational, and broadcast messages. See Chapter 3, "Communicating With Other Users," on page 25 for information about sending and receiving user messages.

Mode Messages

A mode message indicates that the system is ready for a new command or subcommand. The most common mode message is:

```
READY
```

Other mode messages indicate that the system is ready for a valid subcommand. Three commands with subcommands are EDIT, OUTPUT, and TEST. The mode messages associated with these three commands are the command names:

```
EDIT
OUTPUT
TEST
```

For more information about mode messages for subcommands, see *z/OS TSO/E Command Reference*.

Prompting Messages

A prompting message indicates that you need to supply required information or that you supplied incorrect information. Prompting messages include a message identifier (message ID) and are documented in *z/OS TSO/E Messages*. If messages at your terminal do not display a message ID, you can change your profile with the PROFILE MSGID command to display message IDs. For more information about the PROFILE command, see "Changing Your User Profile with the PROFILE Command" on page 155.

Example

The CALL command requires the name of the data set it is to call. If you enter the CALL command without a data set name, the system prompts you for the data set name with a message.

```
READY
CALL
ENTER DATA SET NAME -
```

Respond to prompting messages by typing the requested information and pressing the Enter key.

Example

If you were prompted for a data set name and the data set name was TEST(MYPGM), you would enter after the prompting message:

```
ENTER DATA SET NAME -
test(mypgm)
```

Some messages have additional levels of information available. To see additional information about a message, type a question mark (?) in the leftmost space under the message and press the Enter key.

Example

To see additional information about the following message, enter ?.

```
ENTER DATA SET NAME -
?
```

You then see another message that might be:

```
ENTER DSNAME (MEMBER NAME)
```

If the second message does not contain enough information, you can request another message to give you even more detail.

If you enter a question mark and there are no further messages, you receive the following message:

```
NO INFORMATION AVAILABLE
```

To stop a prompting sequence, enter the requested information or press the attention interrupt key (PA1) to cancel the command. "Interrupting a Process" on page 7 describes the attention interrupt key.

Informational Messages

An informational message tells you about the status of the system or of your terminal session. You do not need to respond to an informational message.

Informational messages include a message identifier (message ID) and are documented in *z/OS TSO/E Messages*. If messages at your terminal do not display a message ID, you can change your profile with the PROFILE MSGID command to display message IDs. For more information about the PROFILE command, see "Changing Your User Profile with the PROFILE Command" on page 155.

Like prompting messages, informational messages may have two or more levels. If an informational message ends with a plus sign (+), you can request an additional message by entering a question mark (?) in the leftmost space under the message as described previously in "Prompting Messages" on page 6.

Broadcast Messages

Broadcast messages are messages that an operator sends using the MVS SEND command or the TSO/E OPERATOR SEND subcommand. The system operator can send messages of general interest to all users of the system or specific messages to individual users.

When an operator sends a message, a console identifier or the characters OPER appear at the end of the message.

Example

An operator might send the following message to all users:

```
DO NOT USE TERMINALS #4, 5 AND 6 ON 6/30.  THEY ARE
RESERVED FOR DEPARTMENT D04.
***
```

When you receive a broadcast message on a display terminal, information on which you were working disappears temporarily from the screen and the broadcast message appears. To return to the information on which you were working, wait for the *** to appear and press the Enter key.

Depending on the logon procedure used at your installation, you might see broadcast messages displayed when you first log on. If messages are not displayed when you log on, you can display broadcast messages at your terminal by issuing the LISTBC command.

You, or any other user, can send messages to other users or to the system operator with the SEND command. For more information on how to send messages, see Chapter 3, "Communicating With Other Users," on page 25.

Interrupting a Process

You can interrupt processing at any time by pressing the key assigned as the attention interrupt key on your terminal. On some terminals such as the IBM 3270 display terminal, the attention interrupt key is labeled "PA1". You can cause an attention interrupt to:

- Terminate processing
- Interrupt processing and resume it again by pressing the Enter key after pressing the attention interrupt key
- Issue the TIME or TEST command during interrupted processing

Terminate processing

You can terminate a program or a process by pressing the attention interrupt key. For instance, if you were executing a program and the program went into a loop, you could press the attention interrupt key to terminate processing.

Example

If you entered a command such as CALL and wanted to get out of the process, press the attention interrupt key. You then see another READY mode message.

```
READY
CALL
 ENTER DATA SET NAME -
 |
 READY
```

Interrupt processing

If, after causing an attention interrupt, you want to resume the operation that you interrupted, press the Enter key before typing anything else. Other keys may cause this same effect on interrupt processing because they simulate pressing the Enter key and entering a null line; however, IBM recommends *not* using other keys. (You can, however, type the TIME or TEST command, which are discussed in the next part.) It is possible that data you were entering or displaying at the time of the attention interrupt might be lost.

If you press the attention interrupt key after issuing a CANCEL, STATUS, or SUBMIT command, the attention interrupt might terminate that command's processing. In this case, pressing the Enter key would not resume command processing.

Note: If you are using Session Manager, press the ERASE EOF key and then press the Enter key to enter a null line to resume execution.

Issue the TIME or TEST command

The TIME and TEST commands can be used after most attention interrupts without terminating the interrupted process.

For example, if you are listing data at your terminal and you want to know what time it is, you can press the attention interrupt key, enter the TIME command, and then press the Enter key again to go back to listing data.

Entering any command other than TIME or TEST causes current processing to terminate. However, you might see one output record from the interrupted program after you enter your next command. This is normal for some programs.

Ending a TSO/E Session

When you are finished using TSO/E, issue the LOGOFF command to sign off from the system and end your terminal session. The system releases your user ID until the next time you issue the LOGON command.

Example

To log off the system, enter at the READY mode message:

```
LOGOFF
```

You can also sign off from the system by issuing the LOGON command. When you sign off using the LOGON command, the system terminates your current session and starts a new one using the options specified on the LOGON command.

Example

To re-log on while you are already logged on, enter the LOGON command at the READY mode message:

```
LOGON
```

You then see a termination message and a prompt for your user ID.

```
YOURID LOGGED OFF TSO AT 13:22:51 ON DECEMBER 10, 1987
ENTER USERID -
```

After you enter your user ID, you see the normal logon procedure used at your installation.

For more information about the LOGON and LOGOFF commands, see *z/OS TSO/E Command Reference*.

Chapter 2. Basic Concepts

Communicating With MVS

TSO/E allows you to communicate with the MVS operating system to do work. Ways to communicate with MVS are through:

- **TSO/E Commands**: Commands control your access to the system, determine your terminal characteristics while you are on the system, allow communication between TSO/E users, and manage data sets.
- **ISPF/PDF Panels**: Interactive System Productivity Facility (ISPF) provides the underlying dialog management service for the ISPF/Program Development Facility (ISPF/PDF). ISPF/PDF is a dialog that allows a TSO/E user to issue TSO/E commands directly or indirectly from panels.

  Information about ISPF/PDF as it relates to specific TSO/E tasks is covered in this book. For more information about ISPF/PDF, see the books listed in the Preface.
- **Programs**: Programs contain instructions that perform tasks. Some common programming languages used under TSO/E are assembler, COBOL, FORTRAN, Pascal, PL/I, REXX, and CLIST. REXX and CLIST (Command List) are high-level interpretive programming languages that allow you to combine TSO/E commands with language statements. For information about CLISTs, see *z/OS TSO/E CLISTs*. For information about REXX, see *z/OS TSO/E REXX User's Guide* and *z/OS TSO/E REXX Reference*.

  For information about programming languages used at your installation, see your installation librarian.
- **JCL**: Job Control Language defines work (jobs) for an operating system. A TSO/E terminal session is considered a job.

  For information about JCL, see *z/OS MVS JCL Reference*.

This book emphasizes how to use TSO/E commands. "Using TSO/E Commands" on page 12 explains:

- The syntax and rules for using TSO/E commands
  - Using command operands
  - Abbreviating commands and keywords
  - Separating words in a command
  - Continuing a command on another line
  - Including comments
  - Subcommands
- How to issue TSO/E commands
- How to get help for TSO/E commands

It also provides a list of common TSO/E commands.

One major function of commands is to manage data sets. Data sets are files or units of information that might contain text, data, programs, or JCL. "Using Data Sets" on page 20 explains:

- Two types of data sets:
  - Sequential
  - Partitioned
- Data set naming rules and conventions
- How to enter data set names

© Copyright IBM Corp. 1988, 2024

Using TSO/E Commands

A TSO/E command consists of a command name that is generally followed by one or more operands. A command name is typically a familiar English word that describes the function of the command; for instance, the RENAME command changes the name of a data set.

Operands provide the command with specific information. For example, the RENAME command has two operands; one that identifies the data set to be renamed and one that specifies the new name.

```
 RENAME     old.data       new.data
/      \   /        \     /        \
command    operand  1     operand  2
```

Two ways to enter commands are:

- Enter the command name and let the system prompt you for required information, or
- Enter the command name and the operands at the same time

Examples of commands in this book, show in uppercase letters the invariable parts of the command such as the command name and operands, and show in lowercase letters the variable parts of the command. When you enter a command, however, you can type the letters as uppercase or lowercase.

Example

To let the system prompt you for required information, enter at the READY mode message:

```
RENAME
```

You then see the message:

```
ENTER OLD DATA SET NAME -
```

After you enter the old data set name, you see another message:

```
ENTER NEW DATA SET NAME -
```

If after you enter the new data set name, the renaming process is successful, you see another READY mode message. If the renaming process is not successful, you see an error message.

For information about prompting, see “Prompting Messages” on page 6.

Using Command Operands

There are two types of operands used with commands: **positional** and **keyword**.

Positional Operands

Positional operands are required and must immediately follow the command name in a certain order. If you enter positional operands incorrectly, you get an error message. For example, the RENAME command requires that the first operand be the old data set name and the second operand be the new data set name.

Example

Switching the operands in a RENAME command results in an error. If OLD.DATA exists and NEW.DATA does not exist, and you type the RENAME command as follows:

```
RENAME new.data old.data
```

You see the message:

```
DATA SET NEW.DATA NOT IN CATALOG OR AMOUNT OF DATASETS
EXCEEDS WORKAREA FOR GENERIC RENAME
READY
```

To enter a positional operand that is a list of several names or values, enclose the list within parentheses and separate the operands by a comma or a space.

Example

To list the data set attributes of more than one data set, enter the LISTDS command with the positional operands, which are the data set names, enclosed in parentheses.

```
LISTDS (new.data test.data)
```

Keyword Operands

Keyword operands are specific names or symbols that have a particular meaning to the system. You can include keywords in any order following the positional operands.

In some cases you specify values with a keyword by entering the value within parentheses following the keyword. Some keywords are not followed by a value within parentheses.

Example

To transmit data set NEW.DATA to USER5 at NODEID, you use a positional operand to specify the destination node and user ID and the keyword operand, DATASET, to specify the data set name. The keyword NOLOG indicates that you do not want a record of the transaction.

```
TRANSMIT   nodeid.user5   DATASET(new.data)   NOLOG
           /          \     /          \     /     \
            positional        keyword       keyword
             operand          operand       operand
```

If you enter mutually exclusive keywords, the last keyword entered overrides the previous ones.

Abbreviating Commands and Keyword Operands

Abbreviating Commands

Nearly all TSO/E commands have abbreviations that you can use in place of the full command name. These abbreviations save you entry time at the terminal. If a command can be abbreviated, it has only one accepted IBM abbreviation. For example, **ALLOC** is the abbreviation for ALLOCATE and **E** is the abbreviation for EDIT. To find the accepted abbreviation for commands used in this book, refer to Table 1 on page 19 or see *z/OS TSO/E Command Reference*.

For readability and clarity in this book, most references to commands and examples of their use appear in the long form.

Abbreviating Keywords

Like commands, nearly all keywords have abbreviations. You may enter keywords spelled exactly as they are shown, or you may use an acceptable abbreviation. An acceptable abbreviation is as much of the keyword as is necessary to distinguish it from the other keywords of the command or subcommand. (If you supply an abbreviation which is not specific enough, the system will prompt you. An exception is the TSO/E HELP command that requires you to enter an operand unambiguously for proper results.)

The LISTBC command, which is used to display messages, has four keywords:

```
MAIL
NOMAIL
NOTICES
NONOTICES
```

The abbreviations are:

```
M for MAIL (also MA and MAI)
NOM for NOMAIL (also NOMA and NOMAI)
NOT for NOTICES (also NOTI, NOTIC, and NOTICE)
NON for NONOTICES (also NONO, NONOT, NONOTI, NONOTIC,
and NONOTICE)
```

Example

To request a listing of the broadcast data set without mail intended for you, you can type the abbreviated version of LISTBC NOMAIL after the READY mode message:

```
LISTB NOM
```

Separating Words in a Command

When you type a command, separate the command name from the first operand by one or more blanks. Separate operands from each other by one or more blanks or a comma.

For example, you can type the LISTBC command like this:

```
LISTBC NOMAIL NONOTICES
```

or

```
LISTBC NOMAIL,NONOTICES
```

or

```
LISTBC NOMAIL     NONOTICES
```

A list of items may be enclosed in parentheses and separated by blanks or commas. For example,

```
LISTDS (mydsa mydsb,mydsc)
```

Continuing a Command on Another Line

When you type a command at the READY message or in ISPF/PDF option 6, you can continue to type beyond the end of the line and the command automatically wraps around to the next line.

However, when you type a command in a CLIST that does not fit on one line, use a plus or minus sign (preceded by a space) as the last character of the first line to continue the command onto the next line. A plus sign removes leading spaces from the continuation line. A minus sign keeps leading spaces with the continuation line.

Example

To display a command to the system as a single line without leading spaces, type:

```
ALLOCATE DATASET(outputds.text) +
             LIKE(old.text)
```

The system sees:

```
ALLOCATE DATASET(outputds.text) LIKE(old.text)
```

To display a command to the system with leading spaces, type:

```
ALLOCATE DATASET(outputds.text) -
       LIKE(old.test)
```

The system sees:

```
ALLOCATE DATASET(outputds.text)       LIKE(old.text)
```

Note: When TSO/E encounters a plus sign or minus sign at the end of a line with no following line, it positions the cursor on the next available input line and waits for you to add a new line.

Including Comments

You can include comments in a TSO/E command anywhere a blank might appear. Comments are most useful in CLISTs that contain TSO/E commands. To include a comment, put it within delimiters /* and */. To continue a comment, use a line continuation character (+ or -) at the end of the line.

Example

A short comment at the end of a line might look like the following:

```
LISTDS (num1.data,num2.data) /* my data sets */
```

A longer comment might need to be split as follows:

```
LISTDS (num1.data,num2.data) /* this is a list of my -
                         active data sets */
```

You do not need to end a comment with */ if the comment is the last thing on the line. Ending a comment with */ is a convention, not a requirement in this case.

Note: To continue a line that contains a comment, use a continuation character *after* the comment:

```
ALLOCATE DATASET(outds.text) /* data set name */ +
         NEW VOLUME(tsomar2)
```

Using Subcommands

Some TSO/E commands such as EDIT, OUTPUT, and TEST have subcommands. Subcommands are similar to commands in that they can have positional and keyword operands.

When you issue a command that has subcommands, the system responds by displaying a special mode message. You can then enter a subcommand to specify the particular operation that you want performed. When you want to stop entering subcommands, enter the END subcommand.

Example

When you issue an EDIT command, instead of READY, the EDIT mode message appears, which indicates that you can enter subcommands of EDIT.

```
  READY
edit test.data emode
 EDIT
list
 00010 THIS LINE IS A TEST.
 00020 THIS LINE IS ALSO A TEST.
 END OF DATA
 EDIT
save
 EDIT
end
 READY
```

In the above example:

- READY and EDIT are mode messages.
- EDIT is also a command. EMODE is an operand of the EDIT command specifying that EDIT should start in EDIT mode, not INPUT mode.
- List, save, and end are EDIT subcommands.

For more information about the EDIT command, see Appendix B, “Using Line Mode Edit,” on page 195. For more information about subcommands, see *z/OS TSO/E Command Reference*.

Issuing TSO/E Commands

You issue TSO/E commands or subcommands by:

- Entering them at your terminal
  - after the READY mode message
  - on the COMMAND/OPTION line of an ISPF/PDF panel and preceded by the characters `tso`.
  - by using the ISPF/PDF COMMAND option (option 6).
- Submitting them in a job using JCL statements. To enter a command using JCL statements, see Chapter 16, “Executing Foreground Commands from a Background Job,” on page 141.
- Including them in a CLIST or REXX exec. For more information about CLISTs, see *z/OS TSO/E CLISTs*. For more information about REXX execs, see *z/OS TSO/E REXX User's Guide* and *z/OS TSO/E REXX Reference*.

Entering more than one command: You can sometimes save time by entering two or more commands separated by field mark symbols (;). For example, if you enter the following ALLOCATE, RENAME, and DELETE commands without waiting for the intervening mode messages, your display is:

```
READY
ALLOCATE DATASET(temp.data) LIKE(other.data) ; RENAME
temp.data new.data ; DELETE temp.data
READY
READY
READY
```

Be careful when entering commands without waiting for the intervening mode messages. If you make a mistake in one of the commands, the system sends you one or more messages, and then cancels the remaining commands you entered. After you correct the error, you have to reenter the other commands. Therefore, unless you are sure your input is correct, wait for a READY message before entering a new command.

Some terminals lock the keyboard after you enter a command, so that when you press the keys, you are not communicating with the system. You cannot enter commands until you get a READY message. Terminals that do not normally lock the keyboard might occasionally do so, for example, when all buffers allocated to the terminal are full. See your terminal operator's guide for information about the terminal you use.

Getting Help for Commands

To display information about any TSO/E command and about some of the command's messages, use the HELP command.

By using the HELP command with its operands, you can request that the system display:

- A list of all IBM-supported TSO/E commands in the system, along with an explanation of each
- Information about a particular command, such as:
  - The function and operation of a command
  - The operands used with a command.
- The syntax for issuing a command
- More information associated with messages that result from executing the VSBASIC, TRANSMIT, or RECEIVE commands.

The EDIT, TEST, and OUTPUT commands have a HELP subcommand. This subcommand acts similarly to and uses similar syntax as the TSO/E HELP command. When using the HELP subcommand, follow the instructions for the HELP command, but substitute the subcommand names.

Listing All TSO/E Commands

To display a list of all the TSO/E commands in the system along with a description of each, enter the HELP command with no operands.

Example

To display all TSO/E commands, enter:

```
HELP
```

You then see information similar to the following:

```
LANGUAGE PROCESSING COMMANDS:

ASM    INVOKE ASSEMBLER F COMPILER.
CALC   INVOKE ITF:PL/1 PROCESSOR FOR DESK CALCULATOR MODE.
COBOL  INVOKE COBOL PROMPTER AND ANS COBOL COMPILER.
FORT   INVOKE FORTRAN PROMPTER AND FORTRAN IV G1 COMPILER.
⋮
```

An installation may also place its own help information about installation-written commands on the system.

Requesting Information about a Command

You can request to see information about a particular command by specifying the HELP command followed by the command name.

Example

To see help information about the RENAME command, enter:

```
HELP RENAME
```

You then see a description of the RENAME command's function, its syntax and operands.

```
FUNCTION -
  THE RENAME COMMAND IS USED TO RENAME A DATA SET OR A
PARTITIONED DATA SET MEMBER OR TO CREATE AN ALIAS FOR A
PARTITIONED DATA SET MEMBER.

SYNTAX -
          RENAME   'DSNAME1' 'DSNAME2'  ALIAS
  REQUIRED - 'DSNAME1' AND 'DSNAME2'
  DEFAULTS - NONE

OPERANDS -
  'DSNAME1' - THE CURRENT DATA SET NAME.
  'DSNAME2' - THE NEW DATA SET NAME TO BE ASSIGNED.
  ALIAS     - MEMBER NAME SPECIFIED BY 'DSNAME2' IS TO BE
              AN ALIAS RATHER THAN A REPLACEMENT.
```

Requesting Specific Information about a Command

To display only the information about a command's function, syntax, or operands, type HELP, and specify the command name followed by the word FUNCTION, SYNTAX, or OPERAND.

Example

To display only the syntax of the RENAME command, enter:

```
HELP RENAME SYNTAX
```

You then see:

```
SYNTAX -
          RENAME   'DSNAME1' 'DSNAME2'  ALIAS
  REQUIRED - 'DSNAME1' AND 'DSNAME2'
  DEFAULTS - NONE
```

Requesting More Information about a Message

To request more information than that provided in the text of a TRANSMIT, RECEIVE, or VSBASIC error message, use the MSGID operand of the HELP command with the message ID enclosed in parentheses.

If messages on your terminal do not display a message ID, see “Changing Your User Profile with the PROFILE Command” on page 155 to read about changing your profile to display message IDs.

Example

To request information about the TRANSMIT message INMX090A, enter:

```
HELP TRANSMIT MSGID(inmx090a)
```

You then see the following:

```
INMX090A  Enter data for receiver. Enter 'string' to stop.

      EXPLANATION:  The TRANSMIT command was invoked with
      the LINE keyword and is prompting you for data or
      message text to be sent.
          .
          .
          .
```

Note: You cannot use the FUNCTION, SYNTAX, OPERANDS, or ALL operands with MSGID.

For more information about the HELP command, see *z/OS TSO/E Command Reference*.

List of TSO/E Commands

Following is a list of the TSO/E commands that are documented in this topic and the major function each command performs.

The commands in this topic are not documented individually. They are documented according to the task each command performs. For an individual description of each command and its operands, see *z/OS TSO/E Command Reference*.

Table 1. List of TSO/E Commands

Command	Abbreviation	Function
ALLOCATE	ALLOC	Allocating data sets.
CALL	CALL	Loading and executing programs.
CANCEL	CANCEL	Halting a submitted job.
DELETE	DEL	Deleting one or more data set entries or one or more members of a partitioned data set.
EDIT	E	Entering data into data sets, or directly modifying data that is already in a data set.
EXEC	EX	Executes a CLIST or REXX exec.
FREE	FREE	Releasing (deallocating) a previously allocated data set.
HELP	H	Obtaining information about the function, syntax, and operands of commands and subcommands and information about certain messages.
LISTALC	LISTA	Listing the data sets that are currently allocated to the TSO/E session.
LISTBC	LISTB	Listing mail and notices for your installation.
LISTCAT	LISTC	Listing the data sets beginning with your prefix or the data sets in a particular catalog.
LISTDS	LISTD	Listing certain attributes of data sets.
LOGOFF	LOGOFF	Ending a terminal session.

Table 1. List of TSO/E Commands (continued)		
Command	**Abbreviation**	**Function**
LOGON	LOGON	Accessing the system.
OUTPUT	OUT	Listing or directing held output.
PRINTDS	PR	Printing a data set on a system printer.
PROFILE	PROF	Listing or changing your user profile.
RECEIVE	RECEIVE	Receiving a transmitted message or data set.
RENAME	REN	Changing the name of a non-VSAM cataloged data set or a member of a partitioned data set, or creating an alias name for a member of a partitioned data set.
RUN	R	Compiling, loading, and executing source statements in a data set.
SEND	SE	Sending messages to another terminal user or the system operator on your system.
SMCOPY	SMC	Copying one data set to another
STATUS	ST	Checking the progress of a job.
SUBMIT	SUB	Submitting a background job for processing.
TERMINAL	TERM	Listing or changing the operating characteristics of your terminal.
TRANSMIT	XMIT	Sending messages or data sets to users on your system or on another system.

Using Data Sets

A data set is a unit of information that can be stored and retrieved.

Some types of data you might put into a data set are:

- A program's source code
- Job control language statements
- Input to a program
- Output from a program
- Memos
- CLIST statements
- Text of a report

A data set is organized in one of several arrangements and is described by control information that the system can access. The two types of data sets most often used with TSO/E are:

- Sequential data set - A single unit with data arranged in a sequence, from top to bottom or beginning to end.
- Partitioned data set (PDS) - Subdivided unit that is divided into separately named, independent partitions called members, each of which can contain data. A partitioned data set has a directory that contains information about each member.

TSO/E Data Set Naming Rules and Conventions

Data Set Naming Rules

Each data set is identified by a unique data set name. When naming data sets, you must follow certain rules:

- A data set name consists of one or more parts connected by periods. Each part is called a qualifier.
- Each qualifier must begin with an alphabetic character (A-Z) or the special characters $, #, @.
- The remaining characters in each qualifier can be alphabetic characters, digits (0-9), a hyphen (-), or the special characters $, #, @.
- Each qualifier must be one to eight characters long.
- The maximum length of a complete data set name before specifying a member name is 44 characters, including the periods.

Example

Some examples of valid data set names are:

```
PARTS
$PARTS.DATA2
A.VERY.LONG.DATASET.NAME.INDEED
```

Data Set Naming Conventions

In addition to the rules, you can make use of certain naming conventions that will make TSO/E easier for you to use. These conventions are an offshoot of the rules. Thus a data set name that follows the naming rules might not follow the naming conventions. The conventions are:

- Data set names consist of three qualifiers.
- The first qualifier of each data set name is your prefix as specified in your user profile. Sometimes your prefix is your user ID.
- The second qualifier of each data set name is your choice; it should be a meaningful name to you.
- The third qualifier is a descriptive qualifier implying certain characteristics of the data. See the following table.

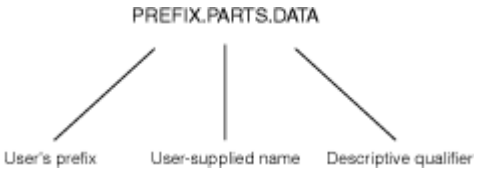
Table 2. Descriptive Qualifiers

Descriptive Qualifier	Data Set Contents
ASM	Assembler (F) input
CLIST	TSO/E commands and CLIST statements
CNTL	JCL and SYSIN for SUBMIT command
COBOL	American National Standard COBOL statements
DATA	Uppercase text
EXEC	TSO/E commands and REXX instructions
FORT	FORTRAN (E, G, GI, H, and GOFORT) statements
LINKLIST	Output listing from linkage editor
LIST	Listings
LOAD	Load module
LOADLIST	Output listing from loader
OBJ	Object module

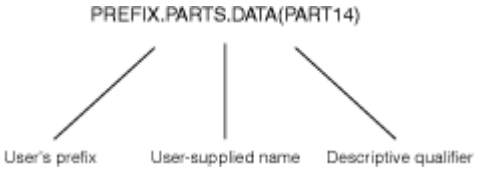
Table 2. Descriptive Qualifiers (continued)

Descriptive Qualifier	Data Set Contents
OUTLIST	Output listing from OUTPUT command
PASCAL	PASCAL statements
PLI	PL/I(F), PL/I Checkout, or PL/I Optimizing compiler statements
TESTLIST	Output listing from TEST command
TEXT	Uppercase and lowercase text
VSBASIC	VSBASIC statements

A data set name that consists of a prefix, a user-supplied name, and a descriptive qualifier is a "fully-qualified" data set name. A fully-qualified data set name looks like:



When you refer to partitioned data sets, enclose the member name in parentheses immediately following the descriptive qualifier. A fully-qualified partitioned data set name looks like:



You do not have to use the conventional descriptive qualifiers when naming a TSO/E data set. However, when a data set name adheres to the conventions, you can refer to the data set by an abbreviated version of the name, and the system supplies the rest of the name.

Example

When you allocate data set PREFIX.OLD.DATA with your prefix, you need only specify the second and third qualifiers because the system assumes your prefix as the first qualifier.

```
ALLOCATE DATASET(old.data) …
```

You must enclose the data set name in single quotation marks if you specify a fully-qualified data set name with a prefix (leftmost qualifier) that is not your own.

Example

When specifying data set PROG.LIST that belongs to a user whose prefix is USER505, type:

```
'user505.prog.list'
```

Entering Data Set Names

The data set naming conventions simplify the use of data set names. If the data set name conforms to the conventions, you need to specify only the user-supplied name field when you refer to the data set in commands.

When you specify only the user-supplied name, the system adds the prefix specified in your user profile, sometimes your user ID and, whenever possible, a descriptive qualifier. The system attempts to derive the descriptive qualifier from available information.

Example

If you want to rename data set PREFIX.TEST5.DATA to PREFIX.TEST4.DATA, rather than specify all of the qualifiers in the RENAME command, you can enter only the middle qualifier:

```
RENAME test5 test4
```

The system adds your prefix as the first qualifier and adds the data set type `data` because that is the data set type assigned to PREFIX.TEST5.DATA.

In some cases, however, the system prompts you for a descriptive qualifier. If you do not specify enough information, the system issues a message at your terminal requesting the required information.

Example

If you want to edit data set PREFIX.SAMPLE.TEXT using line mode edit, you might enter at the READY mode message:

```
EDIT sample
```

The system responds with:

```
ENTER DATA SET TYPE -
```

You then enter the valid data set type:

```
text
```

Data set PREFIX.SAMPLE.TEXT is then available to you and the system responds with the EDIT mode message.

```
EDIT
```

Note: The system will not append qualifiers to any name enclosed in single quotation marks.

Chapter 3. Communicating With Other Users

There are many ways to communicate with other users in TSO/E. You can:

- Use the SEND command to:
  - Send short messages to other users on your system node
  - Send short messages to the master console operator.
  - Send short messages to a specific operator or operator console.
- Use the LISTBC command to display messages sent to you by the system operator and by other users on your system.
- Use the TRANSMIT command to send data sets and longer messages (in data sets) to other users, including users at different nodes
- Set up a Names data set that defines a nickname for a person's *system_node.user_id*
- Use the RECEIVE command to receive transmitted messages from other users
- Access a LOG data set that keeps a record of all TRANSMIT and RECEIVE activity.

Sending Short Messages - SEND Command

The SEND command allows you to send a short message to:

- Another user on your system node
- The master console operator
- A specific operator or operator console (including an extended MCS console).

In order for other users to receive and display the message, their PROFILE must include the INTERCOM operand. For more information about the INTERCOM operand, see “Receiving Messages from Other Users” on page 157.

If the user is logged on and receiving messages, a message sent with the SEND command appears immediately on the user's display screen. If the user is not logged on, the message is deleted unless you specify the LOGON, SAVE, or WAIT operand on the SEND command.

A typical SEND command is as follows:

```
SEND 'This is the message.'  USER(user_id)
```

The message text is enclosed in single quotation marks and must not exceed 115 characters including blanks. To display an apostrophe in the message, type two consecutive single quotation marks. For example when you enter:

```
SEND 'I''m ready.' USER(user_id)
```

the receiver sees,

```
I'm ready.  YOURID
```

For more information about the SEND command, see *z/OS TSO/E Command Reference*.

Sending a Message to Specific Users

USER Operand

To send a message to a specific user, type the SEND command, the message within single quotation marks, and the USER operand with the user ID in parentheses.

© Copyright IBM Corp. 1988, 2024

Example

To send a message to user USER1, enter:

```
SEND 'Don't forget the meeting at 2:00.' USER(user1)
```

To send a message to two or more users, type the SEND command, the message within single quotation marks, and the USER operand with the user IDs separated by commas, and enclosed in parentheses. You can send a message to a maximum of 20 different users.

Example

To send the following message to users USER1 and USER2, enter:

```
SEND 'All you need is on CRSPK1.' USER(user1,user2)
```

To send a message to your own user ID, enter an asterisk instead of a user ID when specifying the USER operand.

Example

To send the following message to your user ID, enter:

```
SEND 'MAP has completed.' USER(*)
```

You then see displayed on your screen:

```
MAP has completed. YOURID
```

Specifying When a User Will Receive a Message

LOGON Operand

To ensure that a user gets the message you send, regardless of whether the user is logged on or is receiving messages, use the LOGON operand. If the user is logged on and receiving messages, the message displays at the user's terminal. If the user is not logged on or is not receiving messages, the system stores the message for the user. The system displays the message at the user's terminal the next time the user issues the LISTBC command or the next time the user logs on and specifies the MAIL operand.

Example

To send the following message to USER1, and to ensure that it is received, regardless of whether USER1 is able to receive the message immediately, enter:

```
SEND 'Did you get FRANK''s new disk?' USER(user1) LOGON
```

SAVE Operand

To send a message to a user, but not have it display at the user's terminal regardless of whether the user is logged on or receiving messages, use the SAVE operand. The system stores the message for the user. The user does not see the message until the next time the user issues the LISTBC command or logs on and specifies MAIL.

Example

To send the following message to USER2, and not have it display until the user either logs on or issues the LISTBC command, enter:

```
SEND 'Extra copies are in my office.' USER(user2) SAVE
```

NOW Operand

NOW is a default operand with the SEND command. If you do not specify the LOGON or SAVE operands with the SEND command, the system immediately displays the message to the user specified. If the user is not logged on or is not receiving messages, the system notifies you and deletes the message.

Note: If your installation uses security labels and security checking, the SEND command may work with restrictions that alter the way the LOGON, SAVE, and NOW operands work on the SEND command. For more information, see “Security Considerations When Sending Messages” on page 28.

Sending a Message to the Master Console Operator

To send a message to the master console operator, enter the SEND command with a message in single quotation marks but omit the user operand. The operator sees the message followed by your user ID.

Example

To send the message "Please cancel my ID" to the master console operator, enter:

```
SEND 'Please cancel my ID'
```

Sending Messages to a Specific Operator or Operator Console

OPERATOR Operand

Specific operators are known to the system by a route code used to identify them. To send a message to a specific operator, use the OPERATOR operand with the operator's specific route code enclosed in parentheses.

Example

To send the following message to the operator identified by route code 7, enter:

```
SEND 'Please mount tapes' OPERATOR(7)
```

If you omit the route code, the default value is 2, which indicates that the system routes the message to the master console operator. If you omit both the USER and OPERATOR operands, the system also routes the message to the master console operator.

CN Operand

You can use the CN operand to send messages to a specific operator console. The console may be a system console or an extended MCS console.

A system console is known to the system by a console identifier or a console name defined by your installation. To send a message to a specific system console, use the CN operand with the console identifier or console name enclosed in parentheses.

An extended MCS console is not associated with a physical device. For example, a user with a TSO/E CONSOLE command authority can act as an extended MCS console. An extended MCS console is known to

the system by a console name (usually the user's user ID). To send a message to a specific extended MCS console, use the CN operand with the console name enclosed in parentheses.

The console name that you specify with the CN operand must be 2 to 8 characters and must begin with an alphabetic character or one of the special characters #, $, or @. The remaining characters may be alphanumeric.

The console identifier that you specify with the CN operand (for system consoles only) must be from 0 to 99. If you specify CN(0), the system sends the message to the master console operator.

Example

To send the following message to the operator console identified by the number 29, enter:

```
SEND 'Did you mount the tapes?' CN(29)
```

Example

To send the following message to the operator console identified by the name TAPE1, enter:

```
SEND 'Please load tape AFD045.' CN(TAPE1)
```

Security Considerations When Sending Messages

Your installation may use various security enhancements to customize how the SEND command works. For example, using RACF®, your installation may control which users can send messages to other users. If you send a message to a user that you are not authorized to send messages to, the system cancels your message and displays an informational message on your terminal.

If your installation uses security labels and security checking, when you use the SEND command to send a message, the security label you are logged on with is associated with the message. The security label on the message has an effect on how the intended receiver of the message receives it.

For example, if you use the SEND NOW command, then even though the intended user is logged on and is receiving messages, the user will not receive the message if the user's security label is less than the message's security label. In this case, the system deletes the message and notifies you.

Also, if you use the SEND LOGON command, and the intended user is logged on and is receiving messages, the user will *not* receive the message if the user's security label is less than the message's security label. In this case, the SEND LOGON request is treated like a SEND SAVE request, and the system stores the message for the user.

With the SEND SAVE command, the system stores the message for the intended user as it does normally. However, because your security label is associated with the stored message, the receiving user will not be able to receive the message later unless the user logs on with a security label equal to or greater than the message's security label. If the receiving user has no such security label defined, the receiving user will never be able to receive your message.

Another point regarding security labels and receiving messages is that the LOGON MAIL command does not display all of your waiting messages if there are mismatches between your security label and the security labels associated with the messages.

Additional security considerations for sending and receiving messages are in “Displaying System Messages - LISTBC Command” on page 28.

Displaying System Messages - LISTBC Command

To retrieve messages that were sent to you by other users or by the system operator, use the LISTBC command. Messages from other users appear as `mail`, while messages from the operator appear as `notices`. The LISTBC command retrieves both mail and notices from the data set where they are stored.

By default, mail and notices are stored in the broadcast data set. However, if your installation uses individual user logs, mail messages are stored in the user log but notices are still stored in the broadcast data set.

If your installation uses security labels and security checking, the LISTBC command may be limited in what it can display for you. LISTBC does not display a stored message unless the security label you are logged on with is equal to or greater than the message's security label. You have the option of logging off and logging back on with a higher security label to view any additional messages.

If you are not authorized to use a security label that is sufficient to receive the stored message, the LISTBC command deletes the message from your user log, and you will never receive it. In this case, *neither you nor the sender of the message is informed by the system that the message has been deleted.*

Example

To display messages stored in a broadcast data set, enter:

```
LISTBC
```

You might then see something like the following:

```
NOTICE .... THE SYSTEM WILL BE DOWN THURSDAY AND FRIDAY
EVENINGS FROM 10 PM TO 6 AM.  IF YOU NEED JOBS PROCESSED
DURING THOSE HOURS, CALL EXT.4444.

Still haven't received your data.  USER2
```

Sending Messages with the TRANSMIT Command

The TRANSMIT command allows you to send the following to users on your system node and on other system nodes:

- A message
- A data set
- Selected members of a partitioned data set
- A message with a data set

A typical TRANSMIT command to transmit a message is as follows:

```
TRANSMIT node_id.user_id
```

The system then prompts you for the message text.

For information on transmitting a data set, see Chapter 10, “Sending and Receiving Data Sets,” on page 91.

Your installation may use various security options to customize how the TRANSMIT command works. Because transmitting a message is essentially the same as transmitting a data set, see “Security Considerations for Sending and Receiving Data Sets” on page 94 for more information about security concerns using the TRANSMIT command.

Transmitting a Message

To transmit a message to another user, type the TRANSMIT command and specify:

- The destination node, followed immediately by a period, and
- The destination user ID

Example

To transmit a message to user USER2 at node NODEID, enter:

```
TRANSMIT nodeid.user2
```

The TERMINAL operand, a default with the TRANSMIT command, causes the system to prompt you to enter the text in either full-screen mode or line mode, depending on the default for the type of terminal you are using.

If your default is full-screen mode, you might type the following:

```
                              DATA FOR NODEID.USER2
00001 Ann,
00002    The meeting scheduled for tomorrow at 1:00 was cancelled.
00003 We're rescheduling it for next Tuesday, same time, same place.
00004    See you there.
00005
```

⋮

```
00021
00022
PF3 = FINISHED|   PF7 = BACKWARD   |   PF8 = FORWARD   |   PA1 = ABORT
```

When you press PF3, the message goes to the user and you see:

```
0 message and 4 data records sent as 5 records to NODEID.USER2
Transmission occurred on 11/02/1987 at 14:07:31.
```

Transmitting to More than One Person

To transmit a message to more than one person, type the TRANSMIT command and:

- Enclose the following in parentheses:
  - The destination node, followed immediately by a period, followed by the destination user ID
  - Separate each `node_id.user_id` by a comma
- Specify the message as described in the previous part

Example

To send a message to USER1 and USER2 at node NODEID enter:

```
TRANSMIT (nodeid.user1,nodeid.user2)
```

For further information on the TRANSMIT command, see *z/OS TSO/E Command Reference*.

Using Nicknames and the Names Data Set

You can reduce repetitive typing of system nodes and user IDs by using nicknames that are defined in the Names data set. The nickname is a 1 to 8-character name that is a synonym for the system node and user ID. The system finds the actual node and user ID by looking up the nickname in the Names data set.

The Names data set allows you to perform the following general functions:

- Abbreviate addresses by associating a nickname with a node name and user ID or with a list of other nicknames (distribution list)
- Provide standard openings and closings for messages
- Control defaults for logging and notification of transmissions

- Use the nicknames section of other Names data sets

The Names data set is composed of two parts:

- The control section. The control section must precede the nicknames section and is terminated by the first nick tag. The control section is used to set defaults for the LOG/NOLOG and NOTIFY/NONOTIFY tags, set prolog or epilog lines, set the default LOG data set name, and identify other Names data sets that the system can use.
- The nicknames section. The nicknames section is a directory of nicknames comprised of one nickname with its supporting data and tags for each entry that you wish to define.

The Names data set in Figure 1 on page 33 contains commonly used tags. Your own Names data set may have these tags as well as any tags listed below.

Tag
Description

:ALTCTL.dsname
The :ALTCTL tag is used in the control section of the PREFIX.NAMES.TEXT data set to provide the name of secondary Names data sets. Your first Names data set must be PREFIX.NAMES.TEXT, but the ALTCTL tags can point to another user's Names data set or another data set that you own. The :ALTCTL tag may appear up to ten times.

:CC.name name ...
The :CC tag is used in a nickname entry to specify a list of users that make up a distribution list. A distribution list can contain up to 100 names. Each name specified may be either a nickname or the name of another distribution list. The explicit user identification (user ID and node name) cannot be used with this tag. The :CC tag is identical to the :LIST tag.

:EPILOG.text
The :EPILOG tag is used in the control section to specify a text line to be appended at the end of any transmitted message. Typically, this will be a standard memo closing. The maximum length of an epilog line is 72 characters. Up to ten epilog lines may be specified.

:LIST.name name ...
The :LIST tag is used in a nickname entry to specify a list of users that make up the distribution list. A distribution list can contain up to 100 names. Each name specified may be either a nickname or the name of another distribution list. The explicit user identification (user ID and node name) can not be used with this tag. The :LIST tag is identical to the :CC tag. If you want to be notified when addressees on a distribution list receive your mail, you must specify :NOTIFY in the distribution list entry or specify NOTIFY(ALL) on the TRANSMIT command.

:LOGSEL.name
The :LOGSEL tag is used in the control section to specify the second (middle) qualifier of all LOG data set names. The value specified with :LOGSEL may be 1 to 8 characters. If no middle qualifier is specified, LOG is used as the middle qualifier (PREFIX.LOG.MISC).

:LOGNAME.name
The :LOGNAME tag is used to specify the third qualifier for the LOG data set name and may be used either in the control section or in a nickname entry. The value specified with the :LOGNAME may be 1 to 8 characters. When used in the control section, the :LOGNAME overrides the default third qualifier for the LOG data set name. When :LOGNAME. is specified in a nickname entry, the value provided overrides both the :LOGNAME set in the control section and the default third qualifier for the LOG data set name. If no :LOGNAME is specified, the default MISC is used (PREFIX.LOG.MISC).

:LOG/:NOLOG
The :LOG and :NOLOG tags control whether a transmit entry appears in a log data set. These operands may be used either in the control section or in the nickname section. When used in the control section, the :LOG or :NOLOG tag controls logging for any addressee specified by node and user ID and also controls logging for any nickname that does not also specify :LOG or :NOLOG. If the nickname entry contains the :LOG or :NOLOG tag, it overrides the value in the control section; but it, in turn, might be overridden by the LOG or NOLOG parameter on the TRANSMIT command.

:LOGLST/:NOLOGLST
The :LOGLST or :NOLOGLST tag is used in a nickname entry that defines a distribution list. The tags indicate whether a log entry should be made for each user in the list (:LOGLST) or not (:NOLOGLST).

:NAME.username
The :NAME tag specifies the name of the user being defined. This name will appear in the copylist and in any log entries for this nickname. The specified name value may be up to 30 characters long.

:NICK.nickname
The :NICK tag is used to begin each nickname entry in the Names data set. It must be the first non-blank (except for line numbers) character of the record. The nickname can be any string of non-blank alphanumeric (A-Z and 0-9) characters and one to eight characters in length. The first :NICK tag separates the control section of a Names data set from the nicknames section.

:NODE.nodeid
The :NODE tag is used within a nickname entry to specify the computer system that you want to TRANSMIT to. If the :NODE tag is not present in a nickname entry, the computer system that you are transmitting from is assumed. This tag can not be included in a distribution list (a nickname entry that includes either a :CC tag or :LIST tag).

:NOTIFY/:NONOTIFY
The :NOTIFY and :NONOTIFY tags can be used either in the control section or in a nickname entry. When used in the control section, the NOTIFY or NONOTIFY tag controls notification for any addressee specified by node and user ID and for any nickname where the nickname entry does not contain :NOTIFY or :NONOTIFY. The value of a :NOTIFY or :NONOTIFY tag can be overridden by the NOTIFY or NONOTIFY parameter on the TRANSMIT command.

Receipt notification is the default for any addressee entered individually on the TRANSMIT command, but not for addressees derived from distribution lists. If you want to be notified for addressees on distribution lists, you must have specified :NOTIFY on the distribution list in the control data set or specified NOTIFY(ALL) on the TRANSMIT command.

Note: If your installation uses security labels and the receiver is at a different security label than you are currently logged on with, you will not be notified when the data is received.

:PARM.text
The :PARM tag can be up to 30 characters of installation-defined data that is passed to RECEIVE command installation exits. For further information about how an installation can use these exits, system programmers can refer to *z/OS TSO/E Customization*.

:PROLOG.text
The :PROLOG tag is used in the control section to specify a text line to be inserted at the beginning of any transmitted message. The maximum length of a prolog line is 72 characters. Up to ten prolog lines may be specified.

:USERID.userid
The :USERID tag identifies the user ID for a nicknamed user. The :USERID tag may not be used in the same entry as the :LIST or :CC tags.

Example of a Names Data Set

Figure 1 on page 33 shows an example of a Names data set.

```
*Control Section
:altctl. OTHER.NAMES.DATA.SET
:prolog. MY NAME
:prolog. MY ADDRESS
:prolog. MY PHONE NUMBER
:epilog. SINCERELY YOURS,
:epilog. MY NAME

*Nicknames Section
:nick.DISTRIB :name. Distribution List
              :list. BILL
                     HOWARD
                     RUTH
                     SUE
                     TOM

:nick.DISTRO :name. Other Distribution List
             :cc. DISTRIB
                  SHERRY
                  MARK
                  JOHN

:nick.DISTRO2 :name. Other Distribution List
              :cc. BILL,DISTRO,ALEX

:nick.TOM :name. Tom Jones
          :node. PLPSC
          :userid. D01TXJ
          :logname. INM.LOG.DATA

:nick.SUE :name. Susan Smith
          :node. ABCXYZ
          :userid. A11SXS

:nick.BILL :name. Bill West
           :node. TDCSYS5
           :userid. H01WWW

:nick.RUTH :name. Ruth Ryan
           :node. ARDNIA
           :userid. T35RYR
           .
           .
```

Figure 1. Example of a Names Data Set

Example

To transmit a message to four of the users in the Names data set, enter:

```
TRANSMIT (tom,sue,bill,ruth)
```

To transmit a message to everyone in the group labeled DISTRIB, enter:

```
TRANSMIT distrib
```

Receiving Transmitted Messages with the RECEIVE Command

The RECEIVE command enables you to receive information that is transmitted to you from the system you are using or from another system. Using the RECEIVE command, you can receive transmitted messages, data sets, or both.

For information about receiving data sets, see “Receiving Data Sets with the RECEIVE Command” on page 93.

Procedures to initiate receiving vary with different installations. Two possibilities are:

- A RECEIVE command is issued automatically when you log on. You then see on your screen the text of all messages sent to you plus system messages that prompt you to receive data sets sent to you.
- A message flashes on your screen when a message or data set is transmitted to you. When you issue the RECEIVE command, you see the text of the message or system messages that prompt you to receive the data set.

If neither of the two above possibilities occur on your system, you might have to enter a RECEIVE command periodically to see if messages and data were sent to you.

What happens when you enter the RECEIVE command depends on whether something was transmitted to you. The possibilities are:

- If nothing was transmitted, the system issues the following message and terminates the process:

```
You have no messages or data sets to receive.
READY
```

- If a message was transmitted, the system displays it at your terminal. For example:

```
Dataset  **  MESSAGE  **  from  USER10 on NODEID
Date:  12 November 1987, 09:58:51 EST
From:  Jon Nolan                          USER10 at NODEID
To:    YOURID  at NODEID

Thanks for responding so promptly.  I'll get the data back
to you by Thursday of next week.
    Jon

Sender notified of receipt
------------------------------------------------------------
No more files remain for the receive command to process.
READY
```

- If several items were transmitted, the system processes them one after another without your having to enter additional RECEIVE commands.

Note: The RECEIVE command may work differently if your installation uses security labels and security checking. See “Security Considerations for Sending and Receiving Data Sets” on page 94 for more information.

Storing Transmitted Data in a Log

A LOG data set is a journal of all TRANSMIT and RECEIVE activity. It keeps a record of:

- What you transmitted
- To whom you transmitted
- When you transmitted
- When what you transmitted was received (acknowledgment)
- What you received
- From whom you received
- When you received

Example of a LOG data set

```
TRANSMIT         ** MESSAGE **              16 SEP 1987 10:26:16
  TO: TOM        NODEID   TJONES  Tom Jones
This is the text of a message that was
transmitted
-------------------------------------------------------------------------
RECEIVE          ** ACKNOWLEDGMENT **       16 SEP 1987 10:34:06
 FROM: TOM       NODEID  TJONES Tom Jones   16 SEP 1987 10:34:06
   STORED        ** MESSAGE **              16 SEP 1987 10:26:16
-------------------------------------------------------------------------
RECEIVE          A.SEQ.DATA                 17 SEP 1987 11:34:06
 FROM: TOM       NODEID JONES Tom Jones     17 SEP 1987 11:30:10
This is the text of a message that was
received along with a data set (SEQ.DATA)
    DSN:  YOURID.SEQ.DATA
```

The three entries in the above LOG data set show:

- A transmitted message
- Acknowledgment that a message you sent was received
- A message received along with a data set

You may have several LOG data sets or just one. The default data set name is usually PREFIX.LOG.MISC but you can specify another name with the :LOGSEL and :LOGNAME tags in your Names data set. The :LOGSEL tag changes the second data set qualifier and the :LOGNAME tag changes the third data set qualifier. For more information about these tags see "Using Nicknames and the Names Data Set" on page 30.

Another way to direct information to a specific log when sending information is with the LOGNAME operand of the TRANSMIT command. The LOGNAME operand changes the third data set qualifier. For more information about this operand, see *z/OS TSO/E Command Reference*.

In addition, you can specify a log name with the LOGDATASET (or LOGDSNAME) operand on either or both of the RECEIVE and TRANSMIT commands. If the log data set does not exist, a sequential data set is created. If the log data set already exists and it is not sequential, you see an error message. This log name operand allows you to keep a log of transmissions.

If your installation uses security labels and security checking, the security label you are logged on at is associated with the transmitted data.

Example

To receive records of transmitted data to a log data set named PREFIX.MY.TRASH, enter:

```
RECEIVE LOGDATASET(my.trash)
```

When you specify more than one log data set, certain operands override others. The possibilities are as follows beginning with the most general. Each successive possibility overrides the previous one.

- No operands - Information goes to PREFIX.LOG.MISC (or whatever default log data set your installation has set up.)
- :LOGSEL and :LOGNAME tags in the control section of your Names data set - Information to and from addressees in the Names data set goes to the specified log data set name whose second and third qualifiers override LOG.MISC.
- :LOGNAME tag in the nickname section of your Names data set - Information to and from a particular addressee in the Names data set goes to the specified log data set name whose third qualifier overrides both the 'MISC' in LOG.MISC and the third qualifier specified with the :LOGNAME tag in the control section of your Names data set.
- LOGNAME operand of the TRANSMIT command - Information sent to a particular person is recorded in a log data set whose third qualifier is specified with the LOGNAME operand.

- LOGDATASET (or LOGDSNAME) operand of the RECEIVE command - Information received is recorded in the log data set specified by the LOGDATASET operand.

The LOG/NOLOG tags and operands of the TRANSMIT command have a similar precedence. When the tags are used in the control section of your Names data set, they can be overridden when they are used with a specific nickname. The LOG/NOLOG operand of the TRANSMIT command overrides all previous specifications.

For more information about these operands, see *z/OS TSO/E Command Reference*.

Part 2. Using Data Sets

Data sets, as mentioned earlier, are files or units of information. By issuing TSO/E commands, you can manage data sets in the following ways:

- Create data sets and specify access to them (ALLOCATE)
- Free data sets from your terminal session (FREE)
- List data set names and information about data sets (LISTALC, LISTCAT, and LISTDS)
- Edit data sets (EDIT)
- Rename data sets (RENAME)
- Copy the data from one data set to another (SMCOPY)
- Send and receive data sets (TRANSMIT, RECEIVE)
- Print data sets (PRINTDS)
- Delete data sets from the system (DELETE)

This part contains the following chapters:

- Chapter 4, “Allocating Data Sets,” on page 39 describes allocation and how to use the ALLOCATE command to access data sets for input and output.
- Chapter 5, “Releasing Data Sets,” on page 57 describes the FREE command and ways you can release allocated data sets.
- Chapter 6, “Listing Data Set Information,” on page 63 describes three commands that list information about data sets. The LISTALC command lists the names and information about data sets currently allocated to your terminal session; the LISTCAT command lists the names and information about data sets that begin with your prefix; the LISTDS command lists information about specific data sets.
- Chapter 7, “Editing Data Sets,” on page 77 describes the TSO/E editing facilities. You can edit data sets using the line mode editor associated with the EDIT command, or you can use the full-screen editor with ISPF/PDF. This chapter briefly describes both methods.
- Chapter 8, “Renaming Data Sets,” on page 81 explains the RENAME command and how you can rename a data set.
- Chapter 9, “Copying Data Sets,” on page 87 explains how you can copy one data set to another using the SMCOPY command.
- Chapter 10, “Sending and Receiving Data Sets,” on page 91 explains how you can send and receive data sets. To send longer messages and data sets, use the TRANSMIT command. To receive longer messages and data sets, use the RECEIVE command.
- Chapter 11, “Printing Data Sets,” on page 95 describes the PRINTDS command and gives examples of the many options for printing and formatting data sets.
- Chapter 12, “Deleting Data Sets,” on page 111 describes how you can use the DELETE command to delete data sets from a particular volume or from the catalog.

© Copyright IBM Corp. 1988, 2024

Chapter 4. Allocating Data Sets

What is Allocation?

Allocation is the process of requesting access to a data set. If you allocate a data set that exists, the system allows you to open the data set. If you allocate a data set that does not exist, the system creates space for it on an available device and allows you to open that space.

A program that requires input and output must be able to open the appropriate data sets before it can run. When a data set is allocated, its resources can be accessed, such as lines of code, input data, or text. You can connect data sets to a program through allocation to a ddname used by the program. If the data set is not connected to a program through allocation to a ddname, the data set should be cataloged.

When running programs in the background, you use job control language (JCL) data definition (DD) statements to allocate the data sets needed for the program. When the job step is over, the data sets are automatically deallocated or freed.

Example of JCL Allocation

```
.
//SYSIN  DD     DSN=IN.DATA,DISP=SHR …
//SYSPRT DD     DSN=OUT.DATA,DISP=(NEW,CATLG) …
.
```

When running programs in the foreground, you can use the ALLOCATE command to allocate the data sets needed for the program. When the program finishes, the data sets remain connected to the program until you free them.

Example of Allocation with the ALLOCATE Command

```
ALLOCATE DATASET(in.data) FILE(sysin) SHR REUSE
ALLOCATE DATASET(out.data) FILE(sysprt) NEW …
```

Security Considerations When Allocating Data Sets

If your installation uses security labels and security checking, the security label you are logged on at is associated with any data set you allocate with a command or procedure. If you log on to a later session with a different security label, you may not be able to access the data set.

Deallocating Data Sets

When you connect data sets to a program in the foreground, the connection remains until you:

- Log off TSO/E - All data sets allocated to your TSO/E session are deallocated when you log off.
- Issue the FREE command - The FREE command releases specified data sets from previous allocations. For information about the FREE command, see “Releasing Data Sets with the FREE Command” on page 57.
- Specify the REUSE operand with an ALLOCATE command - The REUSE operand frees and then reallocates a ddname to a specified data set. For information about the REUSE operand, see “REUSE Operand” on page 42.

To allow other data sets to connect to a ddname, you need to deallocate data sets when a program is finished with them. When you deallocate a data set, the system frees the connection to the ddname, and either deletes the data set or stores it, depending on what was specified when the data set was allocated.

© Copyright IBM Corp. 1988, 2024

Explicit and Implicit Allocation

In a TSO/E session, you can have *explicit* or *implicit* allocation.

- Explicit allocation is a direct request for a data set, which can occur from:
  - Job control language (JCL) - JCL defines a data set as a resource for a job through a data definition (DD) statement. For example, your TSO/E terminal session is a job defined by your LOGON procedure. The LOGON procedure contains JCL DD statements for the data sets that create your TSO/E environment.

    For more information about JCL, see *z/OS MVS JCL Reference*.
  - The ALLOCATE command - You can issue the ALLOCATE command in the foreground from a terminal or from a CLIST. You use the ALLOCATE command to connect a data set to a program for input or output.
  - The ALLOCATE function of the UTILITIES option of ISPF/PDF - You can use it to define a new data set that you can later edit. You cannot, however, use ISPF/PDF to connect a data set to a program.
- Implicit allocation is allocation that occurs indirectly as a result of a command other than the ALLOCATE command. For example, when you issue the EDIT command or select the ISPF/PDF EDIT option, the data set you specify and other data sets needed for the editing process are allocated implicitly. You can then display and edit the data set you specified.

This chapter covers *explicit allocation* using the:

- ALLOCATE command
- ALLOCATE function of the UTILITIES option of ISPF/PDF

Using the ALLOCATE Command

When you use the ALLOCATE command to explicitly allocate a data set or z/OS UNIX file in the foreground, you have a purpose, such as to:

- Provide input to a program from:
  - An existing data set
  - More than one data set
  - Your terminal
- Direct output from a program to:
  - A data set that you specify
  - A system output (SYSOUT) data set
  - Your terminal
- Create a data set

The ALLOCATE command has many operands. Depending on your purpose, you issue the ALLOCATE command with the operands that establish the attributes you want for the data set.

For example, when you create a data set with the ALLOCATE command, you must specify operands that define a new data set or z/OS UNIX file. These operands define the data set name, data set organization, record specifications, and what happens to the data set after it is deallocated.

When the data set exists, you do not need to specify all the operands you used to define a new data set. You specify operands that name the data set, define the type of access, and determine what happens to the data set after it is deallocated.

Providing Input to a Program

When you issue the ALLOCATE command to access a data set you want to use as input to a program, the input data set is connected to the program by a data definition name (ddname) equivalent to the ddnames

used in JCL. To connect a data set to a program, issue the ALLOCATE command and specify the ddname in parentheses after the FILE operand.

You can provide input to a program through:

- An existing data set
- More than one data set
- Your terminal

Accessing an Existing Data Set

To make the connection from a program to an existing data set, issue the ALLOCATE command and the operands that establish the connection you want.

Note: If the existing data set is actually a z/OS UNIX file refer to "Allocating and Accessing UNIX Files" on page 44 for the specific set of operands to use.

Operands Used to Access an Existing Data Set

To access an existing data set, specify operands that:

- Identify the data set - (DATASET operand)
- Identify the ddname, which is the connection to the program - (FILE operand)
- Define the type of access you want - (OLD, SHR, and MOD operands)
- Specify what happens to the data set when it is deallocated - (KEEP and DELETE operands).

In addition, you can specify the REUSE operand to free the ddname from previous allocations.

DATASET Operand

Identify the data set by specifying the data set name enclosed in parentheses after the DATASET operand.

Example

```
ALLOCATE DATASET(input.data) …
```

FILE Operand

The FILE operand specifies the ddname that connects the data set to the program or utility.

Example

```
ALLOCATE DATASET(input.data) FILE(sysin)…
```

OLD, SHR, and MOD Operands

To define the type of access you want for the data set, specify one of the following:

- OLD - gives you exclusive access, such that no other user can access the data set until you are finished using it. You generally request OLD access when you want to modify the data set.
- SHR - gives you non-exclusive access, such that other users can access the data set while you are using it. You generally request SHR access when you want to read data, not modify it.
- MOD - gives you exclusive access to add data to the end of a sequential data set. To add information to a member of a PDS, rewrite the member with the additional records added.

Example

```
ALLOCATE DATASET(input.data) FILE(sysin) OLD
```

KEEP and DELETE Operands

A data set is deallocated when you log off the system or you issue the FREE command. What happens to a data set after it is deallocated is called the disposition of the data set. You can specify the disposition of an existing data set with the following operands:

- KEEP - the system retains the data set. This is the default for existing data sets.
- DELETE - the system deletes the data set.

Example

```
ALLOCATE DATASET(input.data) FILE(sysin) OLD DELETE
```

REUSE Operand

If you want to connect a data set to a program via a ddname and the ddname was previously allocated, you can specify the REUSE operand. The REUSE operand frees the ddname from previous connections before re-allocating.

Example

```
ALLOCATE DATASET(new.data) FILE(sysin) OLD REUSE
```

Accessing More Than One Data Set - Concatenation

You can provide more than one data set as input to a program by concatenating a group of data sets to a ddname. Concatenation means logically linking data sets in a series or chain. You concatenate data sets by issuing the ALLOCATE command as follows:

Example

```
ALLOCATE FILE(indata) DATASET(a.data,b.data,c.data)
```

The search order of data sets within a concatenation is very important. When a program searches through concatenated data sets for a particular item to use as input, it ends its search when it finds the first instance of the item. If correct input is in a member of a partitioned data set and another partitioned data set, earlier in the concatenation, has the same member name, the correct input is not accessed.

Example

You want to execute a CLIST named SETUP that is a member of PREFIX.MY.CLIST and also a member of PREFIX.SYSTEM.CLIST. Both data sets are allocated to file SYSPROC in your LOGON procedure with the following JCL DD statement:

```
//SYSPROC   DD    DSN=PREFIX.SYSTEM.CLIST,DISP=SHR
//                DSN=PREFIX.PROJECT.CLIST,DISP=SHR
//                DSN=PREFIX.MY.CLIST,DISP=SHR
```

You issue the command:

```
EXEC (setup)
```

The member SETUP of the data set PREFIX.SYSTEM.CLIST executes because it appears in the list before the other data set.

You can change the search order of data sets by using the ALLOCATE command and listing the data sets in the order in which you want them searched. Controlling search order is useful when you are developing data and want to have several versions of the same data.

Example

To change the concatenation of the data sets in ddname SYSPROC so that PREFIX.MY.CLIST is searched first, enter the following:

```
ALLOCATE FILE(sysproc) DATASET(my.clist,system.clist,project.clist)
      SHR REUSE
```

The above command deallocates the ddname and then reallocates the data sets in reverse order of the previous example.

Adding a Data Set to a Concatenation

To add a data set to a group of data sets concatenated to a ddname, either:

- Reallocate the entire concatenation including the data set to be added using the ALLOCATE command, and specify the REUSE operand.
- Use the FREE command to free the data sets in the concatenation. Then reallocate the entire concatenation, including the data set to be added, using the ALLOCATE command.

Controlling the block size of concatenated data sets

You can control the block size used to process concatenated data sets by using the BLKSIZE operand to specify one block size for all of the concatenated data sets.

Example

To concatenate data sets T1.DATA, T2.DATA, and T3.DATA within ddname INDATA with block size 8000 for each data set, enter:

```
ALLOCATE FILE(indata) DATASET(t1.data,t2.data,t3.data) +
     BLKSIZE(8000) SHR REUSE
```

When you omit the BLKSIZE operand from the concatenation statement, the block size of the first data set is used for all the data sets. Suppose data sets created with the following block sizes were concatenated as shown in the following example:

```
T1.DATA      BLKSIZE(3200)
T2.DATA      BLKSIZE(8000)
T3.DATA      BLKSIZE(800)
```

Example

```
ALLOCATE FILE(indata) DATASET(t1.data,t2.data,t3.data) SHR REUSE
```

For more information about block size concatenation, see Concatenating PDSs, in *z/OS DFSMS Using Data Sets*.

For more information about operands to use while concatenating data sets, see the ALLOCATE command in *z/OS TSO/E Command Reference*.

Using Your Terminal To Provide Input to a Program

When you are running a program in the foreground, you can use your terminal to provide input to the program by issuing the ALLOCATE command and substituting an asterisk (*) for a data set name.

To use your terminal to provide input to a program that specifies ddname SYSIN as an input data set, enter:

Example

```
ALLOCATE FILE(sysin) DATASET(*)
```

The way to indicate the end of input depends on the program. For example, some programs require that you type /* and others require that you type end.

Allocating and Accessing UNIX Files

You can use the ALLOCATE command to allocate and access UNIX files and the FREE command to release UNIX files. Refer to Chapter 5, “Releasing Data Sets,” on page 57 for information on working with the hierarchical file system on releasing data sets. Refer to *z/OS UNIX System Services User's Guide* for an in-depth discussion of z/OS UNIX, its capabilities, and its advantages. This part is only meant to provide an overview of the facility and its syntax.

Operands Used to Access a UNIX File

To access a UNIX file with the ALLOCATE command, specify operands that provide the:

- UNIX file name - (PATH operand)
- File access and status mode - (PATHOPTS operand)
- File access attributes - (PATHMODE operand)
- Disposition for normal and abnormal file termination - (PATHDISP operand)
- Data set name type - (DSNTYPE operand)
- Type of data - (FILEDATA operand)

Refer to *z/OS TSO/E Command Reference* for allowable options and attributes that you can specify on each of these operands.

PATH Operand

Identify the file by specifying a file name up to 250 characters. Enclose the name in single quotation marks if the name includes characters other than A-Z (upper case), 0-9, or the symbols $, #, @, /, *, +, -, ., &.

Syntax

```
ALLOCATE PATH('pathname')
```

PATHOPTS Operand

Identify the file access and status modes used when accessing the hierarchical file specified on the PATH operand. You can specify file access to specify how the system will manipulate the file.

Syntax

```
ALLOCATE PATH('pathname')
         PATHOPTS(file-option[,file-option,…])
```

PATHMODE Operand

Identify the file access attributes

Syntax

```
ALLOCATE PATH('pathname')
         PATHOPTS(file-option[,file-option,…])
         PATHMODE(file-access-attribute[,file-access-attribute,…])
```

PATHDISP Operand

Identify the file disposition upon normal and abnormal step termination. When coding the PATHDISP operand, you can code either the normal, abnormal, or both disposition types; however, JCL requires you to provide a normal disposition when coding an abnormal disposition.

Syntax

```
ALLOCATE PATH('pathname')
         PATHOPTS(file-option[,file-option,…])
         PATHMODE(file-access-attribute[,file-access-attribute,…])
         PATHDISP(normal-term-dispositon,abnormal-term-disposition)
```

DSNTYPE Operand

Identify the file type as either PIPE (a FIFO file) that is coded in combination with the PATH operand or as an HFS (hierarchical file system) specified in combination with the DSNAME operand. HFS indicates that the data set is an HFS data set which contains a z/OS UNIX hierarchical file system. DSNTYPE does support other values (for example LIBRARY, PDS, EXTREQ, EXTPREF, LARGE, and BASIC), but these are not appropriate for HFS files.

Syntax

```
ALLOCATE PATH('pathname')
         PATHOPTS(file-option[,file-option,...])
         PATHMODE(file-access-attribute[,file-access-attribute,…])
         PATHDISP(normal-term-disposition,abnormal-term-disposition)
         DSNTYPE(PIPE)
```

FILEDATA Operand

When you identify the DD statement as being for a UNIX file, you can also describe the organization of the file so that the system can determine how to process the file. BINARY describes the file as a byte-stream file that does not contain record delimiters. TEXT describes the file as one that contains records delimited

by EBCDIC new line character (X'15'). RECORD describes the file as containing records that may include binary data. See *z/OS DFSMS Using Data Sets*.

Syntax

```
ALLOCATE PATH('pathname')
         PATHOPTS(file-option[,file-option,...])
         PATHMODE(file-access-attribute[,file-access-attribute,…])
         PATHDISP(normal-term-disposition,abnormal-term-disposition)
         DSNTYPE(PIPE)
         FILEDATA(BINARY | TEXT)
```

Example of Allocating and Accessing an UNIX File

To allocate a UNIX file with the following characteristics, code the ALLOCATE command as in the following example:

- Path name including name of file - **/my/open/file.dbp**
- With an access mode - ORDWR (indicating open for read/write)
- Normal termination disposition - KEEP
- Abnormal termination disposition - KEEP
- Regular file
- With access available to the owner for read, write, and execute
- Containing text data.

Example

```
ALLOCATE PATH('/my/open/file.dbp') PATHOPTS(ORDWR,OCREAT)
         PATHDISP(KEEP,KEEP) PATHMODE(SIRWXU) FILEDATA(TEXT)
```

Directing Output From a Program

You can send output from a program, utility, or CLIST to:

- A new or existing data set
- A system output (SYSOUT) data set, which routes it to a printer or other output device
- Your terminal screen

Using a Data Set to Store Output

To make the connection from a program to a data set in which you want to store output, issue the ALLOCATE command, the FILE operand, and other operands that establish the attributes you want for the data set. When you issue the ALLOCATE command to connect a data set to a program, you establish whether it is an input data set or an output data set by the ddname you specify.

For information about the operands used to connect to an existing data set, see “Accessing an Existing Data Set” on page 41. For information about the operands used to connect to a new data set, see “Creating a Non-VSAM Data Set” on page 48.

Using a System Output (SYSOUT) Data Set

A SYSOUT data set is controlled and routed by the system to the job entry subsystem (JES2 or JES3). The job entry subsystem routes the data set to a final output device, based on the class you specify with the SYSOUT operand. After the job entry subsystem routes the data set, the system deletes the data set.

To send output to a system output data set, use the ALLOCATE command with the SYSOUT operand. The SYSOUT operand is mutually exclusive with the NEW, MOD, OLD, and SHR operands. With the SYSOUT operand, you can also specify one of the SYSOUT classes used at your installation. Some output classes

hold the output and do not print it. Others print the output immediately. Others designate when a SYSOUT data set will be printed, either upon deallocation or at the end of a job. Consult your system support center to find out what classes you can use.

SYSOUT Operand

To allocate a data set as a system output data set, issue the ALLOCATE command and specify the FILE operand and the SYSOUT operand followed by the output class enclosed in parentheses. The output class is a single letter defined by your installation.

Example

```
ALLOCATE FILE(sysprt) SYSOUT(a)
```

The ALLOCATE command has operands used specifically with the SYSOUT operand, which control routing and formatting of the data when it is printed. Four of these operands are OUTDES, DEST, SPIN, and SEGMENT.

OUTDES Operand

To specify one or more output descriptors that were created by OUTPUT JCL statements in the LOGON procedure, use the OUTDES operand. Issue the ALLOCATE command and specify the FILE operand, the SYSOUT operand, and the OUTDES operand followed by the names of one or more JCL OUTPUT statements enclosed in parentheses.

Assume your LOGON procedure has OUTPUT JCL statements named PRINTER1 and DIRECTOR.

Example

```
//PRINTER1   OUTPUT  CHAR=GOTH,CLASS=C,COPIES=5
//DIRECTOR   OUTPUT  DEST=NODE2.DIRECTOR
```

To allocate a system output data set defined by ddname REPORT to the associated output descriptors, enter:

Example

```
ALLOCATE FILE(report) SYSOUT OUTDES(printer1,director)
```

For more information about the JCL OUTPUT statement, see *z/OS MVS JCL Reference*.

DEST Operand

To route a system output data set to a specific destination or user at a specific destination, use the DEST operand. Issue the ALLOCATE command and specify the FILE operand, the SYSOUT operand, and the DEST operand followed by the destination enclosed in parentheses.

To route a system output data set defined by ddname MYDATA to user USER3 at node REMOTE1, enter:

Example

```
ALLOCATE FILE(mydata) SYSOUT DEST(remote1.user3)
```

SPIN Operand

To specify when a system output data set is to be available for printing, use the SPIN operand. Issue the ALLOCATE command and specify the FILE operand, the SYSOUT operand, and the SPIN operand followed by the parameter determining when to print.

To make the system output data set (SYSPRT) available for printing upon deallocation, enter:

Example

```
ALLOCATE FILE(sysprt) SYSOUT SPIN(UNALLOC)
```

To make the system output data set (SYSPRT) available for printing at the end of the job, enter:

Example

```
ALLOCATE FILE(sysprt) SYSOUT SPIN(NO)
```

Note:

1. If you specify the SEGMENT operand, it will override the SPIN operand on the ALLOCATE and FREE commands.
2. The SPIN operand specified on the FREE command overrides the SPIN operand specified on the ALLOCATE command.

SEGMENT Operand

Use the SEGMENT operand to specify the number of pages (in an output segment) that the system should write to the system output data set before spinoff processing. Issue the ALLOCATE command and specify the FILE operand, the SYSOUT operand, and the SEGMENT operand followed by the number of pages in parentheses. This specifies how many pages the system will write to the system output data set before spinoff processing.

To specify that the system write a 200-page segment to the system output data set (SYSPRT), enter:

Example

```
ALLOCATE FILE(sysprt) SYSOUT SEGMENT(200)
```

Note: If you specify the SEGMENT operand, it will override the SPIN operand on the ALLOCATE and FREE commands.

For more information about these and other system output operands, see the ALLOCATE command in *z/OS TSO/E Command Reference*.

Using Your Terminal to Display Output From a Program

You can direct output from a program to your terminal by issuing the ALLOCATE command and substituting an asterisk (*) for a data set name.

Assume a program you are running defines an output data set with ddname SYSPRT. To send output from the program to the terminal instead of putting it in a data set, enter:

Example

```
ALLOCATE FILE(sysprt) DATASET(*)
```

Creating a Non-VSAM Data Set

To create a data set, you must define its attributes, for example, its name, type of access, and type of organization. The easiest way to define these attributes is to create a data set like another. You can also define these attributes using specific operands of the ALLOCATE command.

Creating a Data Set Like Another

When you create a data set like another, the attributes of the new data set are the same as the existing data set's. The existing data set does not have to be your own, but if it is a partitioned data set, you must be allowed to access it.

LIKE Operand

To create a data set like another, issue the ALLOCATE command, specify the DATASET operand followed by the new data set name, and specify the LIKE operand followed by the name of the model data set or REFDD operand followed by the DD name of the model data set.

Example

```
ALLOCATE DATASET(test.data) LIKE(model.data)
```

MODEL.DATA then has the same attributes as TEST.DATA.

Creating a Data Set Like Another With Exceptions

Sometimes an existing data set has most of the characteristics you need for a new data set except for one or two. You can allocate the new data set like the existing data set and specify the attributes that you want changed.

Example 1

```
ALLOCATE DATASET(test.data) LIKE(model.data) RECFM(V,B)
```

Example 2

```
ALLOCATE DATASET(test.data) REFDD(maindata) RECFM(V,B,A)
```

Operands Used to Create a Data Set

If you do not create a data set like another, you must specify operands that define the data set attributes. To define a new data set, specify operands that:

- Name the data set - (DATASET operand)
- Identify the ddname, if it is connected to a program - (FILE operand)
- Request access to a new data set - (NEW operand)
- Define the number of directory blocks it needs if it is a partitioned data set - (DIR operand)
- Request storage space for a data set - (BLOCK and SPACE operands)
- Define the organization and record specifications - (DSORG, RECFM, LRECL, BLKSIZE operands)
- Specify what happens to the data set when it is deallocated - (KEEP, CATALOG, and DELETE operands)

For more information about creating data sets, see *z/OS DFSMS Using Data Sets*.

DATASET operand

When you create a data set you specify the DATASET operand followed by a name enclosed in parentheses. The name must be unique and follow the TSO/E data set naming rules specified in “TSO/E Data Set Naming Rules and Conventions” on page 20.

Example

```
ALLOCATE DATASET(new.data) …
```

FILE operand

If you want to connect a new data set to a program to receive output, specify the ddname enclosed in parentheses after the FILE operand. When you create a data set, the FILE operand is optional.

Example

```
ALLOCATE DATASET(new.data) FILE(sysprt) …
```

NEW Operand

When requesting access to a new data set, specify the NEW operand. If you specify MOD instead of NEW, it means to create the data set but if it exists, then you want to append records to it.

Example

```
ALLOCATE DATASET(new.data) FILE(sysprt) NEW …
```

DIR Operand

If the new data set is a partitioned data set, use the DIR operand to specify the number of directory blocks you need for the directory. The directory keeps a list of members and information about each member. Generally you allot one directory block for every six members. Thus a partitioned data set that will have 24 members should have at least 4 directory blocks.

Example

```
ALLOCATE DATASET(new.data) NEW DIR(4) …
```

BLOCK and SPACE Operands

When you allocate a new data set you can specify the amount of storage for the data set. You need to specify space attributes for a new data set only if you do not want to use the system defaults. If the data set is not on DASD, then space attributes have no effect.

SPACE represents the amount of primary space that is initially allocated, followed by the amount of secondary space that is allocated if the primary quantity is not sufficient. The numbers that you specify for SPACE represent one of these quantities:

Tracks of about 48 KB to 56 KB, if you specify TRACKS.
Cylinders of 15 tracks, if you specify CYLINDERS.
Blocks, if you specify BLOCK(nnnn) and not AVGREC.
Records, if you specify BLOCK(nnnn) and AVGREC(U).
Multiples of 1024 records, if you specify BLOCK(nnnn) and AVGREC(K).
Multiples of 1,048,576 records, if you specify BLOCK(nnnn) and AVGREC(M).

Example

```
ALLOCATE DATASET(new.data) NEW DIR(4) BLOCK(800) SPACE(20,10) …
```

DSORG, RECFM, LRECL, and BLKSIZE operands

You can define the organization and record specification of the data set with specific operands as follows. When you omit these operands, the system uses default values. The SMS data class or your compiler or environment might provide different defaults.

- DSORG (Data Set Organization) - specifies the organization of the data set, for example, physical sequential (PS) or partitioned (PO). The default is PS unless you specify a directory count with DIR.
- RECFM (Record Format) - specifies the characteristics of the records in the data set. Specify fixed-length (F), variable-length (V), or undefined-length (U). If you specify F or V, you also can specify B (blocked) and you can specify S, which means "standard" with F or "spanned" with V. With any values, you also can specify either A (ANSI/ISO control characters) or M (machine control characters).
- LRECL (Record Length) - specifies the length in bytes of each record in the data set unless the RECFM value includes U. If the records are variable-length, specify the maximum record length, including a four-byte prefix.
- BLKSIZE (Block Size) - specifies how many bytes are in each data block on the device. If the data set contains fixed-length blocked records, the block size must be a multiple of the record length. For example, if you have a fixed-length blocked record length of 80, the block size can be 3200 (40 x 80) or any other multiple of 80. If the data set contains variable-length records, the block size must be at least 4 bytes greater than the maximum record length. The system supplies an optimized value for BLKSIZE except with RECFM U so do not code BLKSIZE except with undefined-length records. For more information, see *z/OS DFSMS Using Data Sets*.

Example

```
ALLOCATE DATASET(new.data) NEW DIR(4) BLOCK(800) SPACE(20,10)
  DSORG(po) RECFM(f,b) LRECL(80) BLKSIZE(3200)
```

KEEP, CATALOG, and DELETE Operands

To specify what happens to the data set when it is deallocated, use the KEEP, CATALOG, or DELETE operands.

A data set is deallocated when you log off the system or you issue the FREE command. What happens to a data set after it is freed is called the disposition of the data set. You can specify the disposition of any data set with the following operands:

- KEEP specifies that the system retain the data set. If it is SMS-managed (true for most DASD data sets), then this is the same as specifying CATALOG.
- CATALOG specifies that the system store retrievability information about the data set in the system catalog.
- DELETE specifies that the system delete the data set.

Example

```
ALLOCATE DATASET(new.data) NEW DIR(4) BLOCK(800) SPACE(20,10)
  DSORG(po) RECFM(f,b) LRECL(80) BLKSIZE(3200) CATALOG
```

For more information about data set attributes, see *z/OS TSO/E Command Reference*.

Example of Allocating Data Sets to a Utility Program

When you allocate data sets in the foreground, you generally are preparing the environment for a program that you are going to run in the foreground. Earlier examples in this chapter show isolated ALLOCATE commands, but not how you use those commands together to prepare the environment for a program.

The following example uses IEBGENER, a data set utility program provided by IBM. Data set utility programs are used to reorganize, change, or compare data in data sets. Your installation might have other programs that do similar functions.

One of the things you can do with IEBGENER is convert sequential data to a partitioned data set with a specified number of members. If you have a sequential data set and you want to convert the data set to a partitioned data set with three members, do the following steps:

1. In the sequential data set, note the record that marks the end of each member you intend to create, or insert a line with a unique word.
2. Issue four ALLOCATE commands to set up the environment for IEBGENER.
3. Run IEBGENER with the CALL command.

Assume you have the following information:

- Message data set used by IEBGENER, which you want displayed at your terminal
  - Ddname - SYSPRINT
  - Data set name - *
- Sequential data set used as input
  - Name - PREFIX.LINES.DATA
  - Ddname - SYSUT1
  - Access - OLD
  - Data set disposition - KEEP
- Partitioned data set used as output
  - Data set name - PREFIX.NEWSET.DATA
  - Ddname - SYSUT2
  - Access - NEW
  - Disposition - CATALOG
  - Directory blocks - 3
  - Data set organization - PO
  - Blocks - 3200 bytes
  - Space
    - Primary unit = 20
    - Secondary unit = 10
- Record format - fixed, blocked
- Logical record length - 80 bytes
- Block size - 3200
- Input data set, which is your terminal
  - Ddname - SYSIN
  - Data set name - *
- New member information given as input
  - Member names - ONE, TWO, THREE
  - Last record information for first member
    - length (7 bytes)
    - name of last record in first member (ONEDONE)
    - column where the name begins (column 1)
  - Last record information for second member

- length (7 bytes)
- name of last record in second member (TWODONE)
- column where the name begins (column 1)

The first ALLOCATE command defines that the message data set be sent to your terminal.

Example

```
ALLOCATE FILE(sysprint) DA(*) REUSE
```

The second ALLOCATE command defines the sequential data set to be used as input.

Example

```
ALLOCATE DATASET(lines.data) FILE(sysut1) OLD KEEP
```

The third ALLOCATE command defines the partitioned data set to be produced as output.

Example

```
ALLOCATE DATASET(newset.data) FILE(sysut2) NEW CATALOG DIR(3) DSORG(PO)
  BLOCK(3200) SPACE(20,10) RECFM(f,b) LRECL(80) BLKSIZE(3200)
```

The fourth ALLOCATE command defines your terminal as the source of input.

Example

```
ALLOCATE FILE(sysin) DATASET(*)
```

After all the resources have been allocated, you can call the utility program as follows:

Example

```
CALL 'SYS1.LINKLIB(IEBGENER)'
```

Because you allocated the message data set to your terminal, you are prompted with messages that guide you to input the following highlighted information. When you press the Enter key, information you typed is repeated.

Example

```
DATA SET UTILITY - GENERATE
                 PAGE 0001
       GENERATE    MAXNAME=3,MAXGPS=2   /* 3 members and 2 stopping points
         MEMBER    NAME=ONE             /* First member is named one
 GROUP1  RECORD    IDENT=(7,'ONEDONE',1)/* Last record in one is onedone
         MEMBER    NAME=TWO             /* Second member is named two
 GROUP2  RECORD    IDENT=(7,'TWODONE',1)/* Last record in two is twodone
         MEMBER    NAME=THREE           /* Third member is named three
/*
PROCESSING ENDED AT EOD
```

Using ISPF/PDF to Allocate Data Sets

Allocation in ISPF/PDF is different from allocation with the ALLOCATE command. In ISPF/PDF you use the ALLOCATE function to create a data set. Before a data set is created in ISPF/PDF, it cannot be accessed.

You can allocate a new data set while in ISPF/PDF by entering attribute information on panels. You cannot, however, associate the data set with a ddname. To allocate a data set and associate it with a ddname, use the ALLOCATE command.

To allocate a new data set, select the UTILITIES option (option 3) on the ISPF/PDF Primary Option Menu.

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 3
                                                          USERID   - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME     - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS  - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

You then see the Utilities Selection Menu, which allows you to do many data set management tasks. To allocate a new data set, select the DATASET option (option 2).

```
--------------------------  UTILITY SELECTION MENU  ----------------------------
OPTION ===> 2

    1  LIBRARY     - Compress or print data set.  Print index listing.
                     Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                     display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                     Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

The next panel you see, the Data Set Utility panel, allows you to specify an action and a data set name. To allocate a data set, type A on the OPTION line. Then specify the data set name in the three ISPF LIBRARY fields or in the DATA SET NAME field under OTHER PARTITIONED OR SEQUENTIAL DATA SET.

Specifying a Data Set Name

The ISPF LIBRARY fields are most often used for data sets that have three qualifiers and follow the data set naming conventions. The OTHER PARTITIONED OR SEQUENTIAL DATA SET field is similar to data set specification in line mode TSO/E, and generally is used for a data set with fewer than or more than three qualifiers. For example, to specify data set PROJECT.ONE, which does not begin with your prefix, type:

```
OTHER PARTITIONED OR SEQUENTIAL DATA SET
    DATA SET NAME ===> 'project.one'
```

To specify data set PREFIX.FIRST.PROJECT.MEMO, you can eliminate the prefix and the single quotation marks and type:

```
 OTHER PARTITIONED OR SEQUENTIAL DATA SET
     DATA SET NAME ===> first.project.memo
```

For more information about naming and specifying data sets, refer to "TSO/E Data Set Naming Rules and Conventions" on page 20.

```
---------------------------------DATA SET UTILITY -----------------------------
OPTION ===> a

    A - Allocate new data set                 C - Catalog data set
    R - Rename entire data set                U - Uncatalog data set
    D - Delete entire data set                S - Data set information (short)
    blank - Data set information

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  test
    TYPE    ===>  data

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>          (If not cataloged, required for option "C")

DATA SET PASSWORD ===>          (If password protected)
```

When you press the Enter key on the Data Set Utility panel, you see the Allocate New Data Set panel with various data set attributes as data entry fields. What you type in these fields depends on what you want the attributes to be and whether or not the data set is sequential or partitioned. Generally, the information that appears in the fields is left from the last allocated data set.

To see descriptions of the data entry fields, press the HELP PF key while the Allocate New Data Set panel is displayed. When you finish typing information in the fields, press the Enter key to create the data set. The following example is for allocating a sequential data set.

```
----------------------------  ALLOCATE NEW DATA SET  ---------------------------
COMMAND ===>

DATA SET NAME:  PREFIX.TEST.DATA

   VOLUME SERIAL     ===>                (Blank for authorized default volume)*
   GENERIC UNIT      ===>                (Generic group name or unit address)*
   SPACE UNITS       ===> blks           (BLKS, TRKS or CYLS)
   PRIMARY QUAN      ===> 20             (in above units)
   SECONDARY QUAN    ===> 10             (in above units)
   DIRECTORY BLOCKS  ===> 0              (Zero for sequential data set)
   RECORD FORMAT     ===> FB
   RECORD LENGTH     ===> 80
   BLOCK SIZE        ===> 3120
   EXPIRATION DATE   ===>                (YY/MM/DD
                                          YY.DDD in julian form
                                          DDDD for retention period in days
                                          or blank)


   ( * Only one of these fields may be specified)
```

When you press the Enter key on the Allocate New Data Set panel, the data set is created and you see this message in the upper right hand corner of the panel:

```
DATA SET ALLOCATED
```

An example of allocation for a partitioned data set is as follows:

```
----------------------------  ALLOCATE NEW DATA SET  ---------------------------
COMMAND ===>

DATA SET NAME:  PREFIX.TEST.DATA

   VOLUME SERIAL     ===>                (Blank for authorized default volume)*
   GENERIC UNIT      ===>                (Generic group name or unit address)*
   SPACE UNITS       ===> blks           (BLKS, TRKS or CYLS)
   PRIMARY QUAN      ===> 50             (in above units)
   SECONDARY QUAN    ===> 20             (in above units)
   DIRECTORY BLOCKS  ===> 2              (Zero for sequential data set)
   RECORD FORMAT     ===> FB
   RECORD LENGTH     ===> 80
   BLOCK SIZE        ===> 8000
   EXPIRATION DATE   ===>                (YY/MM/DD
                                          YY.DDD in julian form
                                          DDDD for retention period in days
                                          or blank)


   ( * Only one of these fields may be specified)
```

For more information about allocating data sets in ISPF/PDF, see *z/OS ISPF User's Guide Vol I*.

Chapter 5. Releasing Data Sets

The FREE command releases (deallocates) data sets and z/OS UNIX files that were previously allocated to programs in your TSO/E session and makes those data sets available to other programs. The FREE command also releases ddnames that link input and output data sets to programs, utilities, and CLISTs.

There is a maximum number of data sets that can be allocated at the same time in your terminal session. Data sets that are implicitly allocated are automatically released when you are finished with them. However, data sets that you explicitly allocate with the ALLOCATE command, remain allocated until you either free them with the FREE command or log off the system.

To avoid reaching the maximum limit of allocated data sets and being denied access to desired resources, periodically use the FREE command to release resources that you no longer need.

Releasing Data Sets with the FREE Command

Using the FREE command, you can request that the system:

- Release all data sets and ddnames not currently in use, which were explicitly allocated with the ALLOCATE command (ALL operand)
- Release one or more specific data sets (DATASET operand)
- Release a z/OS UNIX file (PATH operand)
- Release a ddname from previous allocations (FILE operand)
- Send SYSOUT data sets to a specific printer or destination (DEST operand)
- Release the SYSOUT data set for printing after deallocation or at the end of the job (SPIN operand)
- Place a data set in a HOLD queue to print later or view only (HOLD operand)
- Not place a data set in a HOLD queue (NOHOLD operand)
- Retain the data set after you release it (KEEP operand)
- Delete the data set after you release it (DELETE operand)
- Retain the data set in a catalog after you release it (CATALOG operand)
- Remove the data set from a catalog after you release it. The system still retains the data set (UNCATALOG operand).

Releasing All Your Data Sets Not Currently in Use

ALL Operand

Use the ALL operand to release all data sets and ddnames that were allocated to your user ID with the ALLOCATE command, but are not currently in use.

Example
A data set allocated to your user ID is tying up the system, and you are not sure which data set is causing the problem. You decide to free all your data sets until you can correct the problem. You enter: `FREE ALL`

Note: After issuing FREE ALL, you may be unable to use ISPF/PDF or other programs that are dependent on the data sets just released. To regain access to ISPF/PDF and other programs, log off and log on again.

© Copyright IBM Corp. 1988, 2024

Releasing Specific Data Sets

You can release previous allocations of a:

- Data set
- Group of data sets
- Ddname associated with specific data sets
- Group of ddnames associated with specific data sets

DATASET (or DSNAME) Operand

To release a specific data set from all previous allocations, type the FREE command:

- Specify the DATASET or DSNAME operand.
- Enclose the data set name in parentheses.
  - You must include the descriptive qualifier in the data set name.
  - You may include a member name in parentheses when specifying a partitioned data set.

Example

To release data set PREFIX.PARTS.DATA, enter:

```
FREE DATASET(parts.data)
```

To release a group of specific data sets from previous allocations, type the FREE command:

- Specify the DATASET or DSNAME operand
- Enclose in parentheses the data set names separated by commas

Example

To release data sets PREFIX.PARTS.DATA, PREFIX.REPORT.TEXT, and PREFIX.SIMIAN.CNTL(HAPE), enter:

```
FREE DATASET(parts.data,report.text,simian.cntl(hape))
```

FILE (or DDNAME) Operand

To release the data set(s) associated with a specific ddname, type the FREE command and specify:

- The FILE or DDNAME operand
- The ddname name enclosed in parentheses

Example

To release the data set(s) associated with file SYSUT1, enter:

```
FREE FILE(sysut1)
```

To release the data sets associated with a specific group of ddnames, type the FREE command and specify:

- The FILE or DDNAME operand
- The ddnames enclosed in parentheses

Example

To release the data set associated with ddnames SYSUT1, SYSUT3, SYSPRINT, and SYSIN, enter:

```
FREE FILE(sysut1,sysut3,sysprint,sysin)
```

Releasing UNIX Files

You can use the FREE command to release UNIX files. To allow this support, both JCL and TSO/E support a pair of keywords on FREE. These two keywords are a subset of keywords used on the ALLOCATE command to allocate and access UNIX files. Refer to *z/OS UNIX System Services User's Guide* for an in-depth discussion of z/OS UNIX System Services, its capabilities, and its advantages. This part is only meant to provide an overview of the facility and its syntax.

Operands Used to Release a UNIX File

To release a UNIX file, specify operands that provide the:

- Hierarchical file name - (PATH operand)
- Disposition - (PATHDISP operand)

Refer to *z/OS TSO/E Command Reference* for allowable options and attributes that you can specify on each of these operands.

PATH Operand

Identify the file by specifying a file name up to 250 characters. Enclose the name in single quotation marks if the name includes characters other than A–Z (upper case), 0–9, or the symbols $, #, @, /, *, +, -, ., &.

Syntax

```
FREE PATH('pathname')
```

PATHDISP Operand

Identify the file disposition.

Syntax

```
FREE PATH('pathname')
     PATHDISP(normal-term-disposition)
```

Example of Releasing a UNIX File

To release a UNIX file with the following characteristics, overriding the PATHDISP specified at allocation, code the FREE command as in the example:

- Named - **/u/userid/file.dbp**
- Disposition - DELETE

Example

```
FREE PATH('/u/userid/file.dbp') PATHDISP(DELETE)
```

The operating system accepts the request to delete the file. If there are multiple names, the operating system will not actually delete the file.

Releasing SYSOUT Data Sets and Sending Them to a Location

DEST Operand

To release a SYSOUT data set and send it to another location for output processing, enter the FREE command and specify:

- The SYSOUT data set and the associated file name
- The DEST parameter with the node ID of the location enclosed in parentheses

Example

To release the SYSOUT data set associated with the file name SYSPRINT and send it to location LONDONW3, enter:

```
FREE FILE(sysprint) SYSOUT DEST(londonw3)
```

You can send multiple SYSOUT data sets to a location by following the same procedure for releasing multiple data sets.

If you omit the DEST operand when releasing SYSOUT data sets, the data sets are directed to the location you specified at the time you allocated the data set.

Releasing SYSOUT Data Sets for Printing

SPIN Operand

To release a SYSOUT data set for printing upon deallocation or at the end of the job, enter the FREE command and specify:

- The SYSOUT data set and the associated file name
- The SPIN parameter that specifies when the SYSOUT data set is available for printing.

When you specify the operand, you must also specify either of the following parameters:

UNALLOC
For printing upon deallocation

NO
For printing at the end of the job.

Example

To release the SYSOUT data set associated with the file name SYSPRINT and send it to location LONDONW3, enter:

```
FREE FILE(sysprint) SYSOUT DEST(londonw3)
```

If you specify the SPIN operand without entering a parameter, the FREE command prompts you for one.

If you omit the SPIN operand, the data set will be released at the end of the job.

The SEGMENT operand specified on the ALLOCATE command overrides the SPIN operand on the FREE command.

The SPIN operand specified on the FREE command overrides the SPIN operand on the ALLOCATE command.

For more information about these and other system output operands, see the ALLOCATE and FREE commands in the *z/OS TSO/E Command Reference*.

Releasing Data Sets and Placing Them in a Hold Queue

HOLD Operand

You can send data to the hold queue if you want to view it with the option of printing it later. To place a data set on a hold queue, type the FREE command and specify:

- The data set name
- The HOLD operand

Example

To place the data set associated with file SYSUT1 in a hold queue, enter:

```
FREE FILE(sysut1) HOLD
```

When you specify the HOLD operand, it overrides any HOLD/NOHOLD specification made when the data set was originally allocated, and it also overrides the default HOLD/NOHOLD specification associated with a SYSOUT class.

NOHOLD Operand

To ensure that the system not place a data set on a hold queue, type the FREE command and specify:

- The data set name
- The NOHOLD operand

Example

To keep the data set associated with file SYSUT1 from being placed on a hold queue, enter:

```
FREE FILE(sysut1) NOHOLD
```

Releasing Data Sets and Specifying Their Disposition

When you release a data set, you can also tell the system what you want done with the data set after it is released. You can specify one of four data set dispositions:

- KEEP
- DELETE
- CATALOG
- UNCATALOG

KEEP Operand

To retain a data set after you release it, type the FREE command and specify:

- The data set name or file name
- The KEEP operand

Example

To retain file SYSPRINT in the system after you release it, enter:

```
FREE FILE(sysprint) KEEP
```

DELETE Operand

To delete a data set after you release it, type the FREE command and specify:

- The data set name
- The DELETE operand

Example

To delete data set PARTS.DATA from the system after you release it, enter:

```
FREE DSNAME(parts.data) DELETE
```

When releasing a data set that was allocated with a disposition of SHR, or when releasing a member of a partitioned data set, the DELETE operand is not valid.

When you release a SYSOUT data set, the DELETE operand is the only valid disposition operand. It deletes the data set without printing it.

Note: If you specify FREE FILE(SYSUT2) DELETE, all the data sets associated with ddname SYSUT2 are deleted.

CATALOG Operand

To retain a data set in a catalog after you release it, type the FREE operand and specify:

- The data set name
- The CATALOG operand

Example

To retain the data set(s) associated with file SYSUT2 in a catalog after you release it, enter:

```
FREE FILE(sysut2) CATALOG
```

UNCATALOG Operand

To remove a data set from a catalog after you release it, type the FREE command and specify:

- The data set name
- The UNCATALOG operand

Example

To remove the data set associated with file SYSUT2 from a catalog after you release it, enter:

```
FREE FILE(sysut2) UNCATALOG
```

For more information about the FREE command, see *z/OS TSO/E Command Reference*.

Chapter 6. Listing Data Set Information

You can request that the system display various types of information about data sets such as:

- A list of data sets currently allocated to your terminal session. (LISTALC command)
- A list of data sets with your prefix as the first qualifier, or a list of data sets from a particular catalog. (LISTCAT command or ISPF/PDF UTILITIES option)
- Data set attributes such as the record format, record length, block size, and data set organization. (LISTDS command or ISPF/PDF UTILITIES option).

In addition, you can display the contents of a data set by using:

- **ISPF/PDF BROWSE or EDIT** - You can display data set contents on a panel and scroll backward and forward through the contents. For more information about ISPF/PDF EDIT, see Chapter 7, “Editing Data Sets,” on page 77 or *z/OS ISPF User's Guide Vol I*.
- **LIST subcommand of EDIT** - While you are in EDIT mode, you can enter the LIST subcommand to display the contents of the data set you are editing. For more information, see Appendix B, “Using Line Mode Edit,” on page 195.

Listing Allocated Data Sets - LISTALC Command

Use the LISTALC command to list the names of data sets currently allocated to your prefix during the TSO/E session. You can also list attributes of these data sets by specifying operands with the LISTALC command. The list can include the data sets that were dynamically allocated and any data sets previously allocated for temporary use by the ALLOCATE command.

Using the LISTALC command you can display:

- The data sets currently allocated to your prefix during the TSO/E session
- The data definition name (ddname) associated with a data set (STATUS operand)
- The disposition of a data set (STATUS operand)
- The history of a data set (HISTORY operand). The history of a data set includes:
  - The date the data set was created
  - The date the data set will expire (retention period)
  - Whether the data set is protected
  - The organization of the data set (sequential or partitioned).
- The names of the members of a partitioned data set (MEMBERS operand)
- The fully-qualified name of a system-generated data set name (SYSNAMES operand)

When you see the list, data sets allocated to your terminal are followed by the word **TERMFILE**. If an asterisk precedes the data set name in the list, the data set is allocated but not currently in use.

Listing the Data Sets Allocated to Your User ID

There are times when you might want to find out exactly what data sets the system has currently allocated to your user ID. To list all data sets allocated to your user ID during the current TSO/E session, enter the LISTALC command without operands.

© Copyright IBM Corp. 1988, 2024

Example

To see a list of data sets currently allocated to your user ID and a list of data sets associated with a ddname, enter:

```
LISTALC
```

You might see something like the following:

```
SYS1.TSO.CLIST
ISP.ISPF.CLISTS
CATALOG.VTSO022
SYS1.HELP
TERMFILE
PREFIX.ISPF.PROFILE
⋮
```

Displaying Ddnames and Data Set Disposition

STATUS Operand

When writing programs and JCL statements, it is often useful to know if other programs use the same data sets you do, and when and how they intend to do so. You can list this type of information by specifying the STATUS operand. In addition to displaying the data set name, the STATUS operand displays the data definition names (ddnames or files), and the data set's disposition.

A data set's disposition is indicated by one of the following operands:

CATLG
requests that the system retain the data set and enter the data set's name into the system catalog.

UNCATLG
requests that the system remove the data set's name from the system catalog, but still retain the data set.

KEEP
requests that the system retain the data set.

DELETE
requests that the system remove the data set's name from the system catalog and release the storage space the data set formerly occupied.

Example

To list the ddnames and dispositions associated with each data set currently allocated to your user ID, enter:

```
LISTALC STATUS
```

You then see something like the following:

```
SYS1.TSO.CLIST
  SYSPROC  KEEP
ISP.ISPF.CLISTS
           KEEP
CATALOG.VTSO022
  SYS00002 KEEP,KEEP
SYS1.HELP
  SYSHELP  KEEP
TERMFILE SYSPRINT
SYS1.ICQGCTAB
  ICQGCTAB KEEP
NULLFILE  $CNTL
PREFIX.ISPF.PROFILE
  ISPPROF  KEEP
⋮
```

When no ddname appears under a data set name, that data set was not allocated to a file. When no ddname appears but a disposition appears, the data set was allocated to the file that was specified above it. For example, in the above list, ISP.ISPF.CLISTS was allocated to file SYSPROC.

Listing the History of a Data Set

HISTORY Operand

Before you access a data set, you might want to know more about it. For example, you might want to know when the data set was created, or when it will expire (be removed from the system). To avoid errors, you would also want to know if the data set is password-protected and how the data set is organized.

To list this kind of information, use the HISTORY operand. The system then displays information under the headings:

```
--DSORG--CREATED-----EXPIRES-----SECURITY
```

The possible values for DSORG and their meanings are:

DSORG
Meaning

PS
A sequential data set

PO
A partitioned data set

IS
An indexed sequential data set

DA
A direct access data set

VSAM
A virtual storage access method (VSAM) data set

DIR
z/OS UNIX Directory

CSPEC
z/OS UNIX character special file

FILE
z/OS UNIX regular file

FIFO
z/OS UNIX FIFO special file

SYMLK
z/OS UNIX symbolic link

TAPE
A data set on tape

******
An unspecified type of data set

??
Any other type of data set

PSU
An unmovable sequential data set

POU
An unmovable partitioned data set

Example

To find out each data set's organization, creation and expiration dates, and whether the data set has security, enter:

```
LISTALC HISTORY
```

You then see something like the following:

```
--DSORG--CREATED-----EXPIRES-----SECURITY
SYS1.TSO.CLIST
  PO     07/29/1987  00/00/0000   RACF
ISP.ISPF.CLISTS
  PO     08/05/1987  00/00/0000   NONE
CATALOG.VTSO022
  **     06/24/1986  12/31/1999   RPWD
SYS1.HELP
  PO     01/31/1985  00/00/0000   RACF
TERMFILE
SYS1.ICQGCTAB
  PO     06/04/1986  00/00/0000   NONE
NULLFILE
PREFIX.ISPF.PROFILE
  PO     09/21/1985  00/00/0000   NONE
⋮
```

Listing the System-generated Data Set Names

SYSNAMES Operand

To list the fully-qualified system-generated data set names for each data set currently allocated to your user ID, specify the SYSNAMES operand.

For more information about the LISTALC command, see *z/OS TSO/E Command Reference*.

Listing Catalog Information - LISTCAT Command

Use the LISTCAT command to obtain a list of data sets with your prefix as the first qualifier, or to obtain a specific list of data sets and data set information from a catalog. A catalog contains information about data sets, such as who created it, when it was created, when it will expire, and the serial number of the volume in which the data set is stored.

Using the LISTCAT command you can request that the system:

- List all the data sets beginning with your prefix (no operands)
- List the data sets of a particular catalog (CATALOG operand)
- Put the list information in a data set rather than on the terminal screen (CATALOG and OUTFILE operands)
- List the information associated with only specific data sets (ENTRIES operand)
- List the data sets with a particular qualifier (LEVEL operand)
- List data sets with aliases only (ALIAS operand)
- List the data sets that were created a specific number of days before the present (CREATION operand)
- List the data sets that expire within a certain number of days (EXPIRATION operand)
- Return all or a part of the following information about data sets in your personal library (ALL operand):
  - Data set name
  - Owner identification
  - Creation date
  - Expiration date
  - Entry type
  - Volume serial number
  - Device types

For information about how to list catalog information for VSAM data sets, see *z/OS DFSMS Access Method Services Commands*.

When you enter LISTCAT with no operands, the last prefix specified with the PROFILE command becomes the highest-level data set name qualifier. The system displays only information about the data sets associated with that prefix.

Listing Data Sets With Your Prefix

To display a list of all the data sets with your prefix as the first qualifier, use the LISTCAT command with no operands.

Example

To display a list of data sets whose names begin with your prefix, enter:

```
LISTCAT
```

You then see something like the following:

```
PREFIX.DATA.DATA
PREFIX.EXAMPLE.TEXT
PREFIX.LOG.MISC
PREFIX.MEMO.TEXT
PREFIX.MY.DATA
PREFIX.MY.NOTE
PREFIX.TEST1.DATA
PREFIX.TEST2.DATA
⋮
```

Listing Information About Specific Data Sets

ENTRIES Operand

You can list the catalog information about specific data sets using the ENTRIES operand. To list specific data set information, specify the ENTRIES operand followed by the data set name(s) enclosed in parentheses.

Example

To see catalog information about data set PREFIX.EXAMPLE.TEXT, enter:

```
LISTCAT ENTRIES(example.text)
```

You then see something like the following:

```
NONVSAM ------- PREFIX.EXAMPLE.TEXT
     IN-CAT --- CATALOG.VTSO005
```

Listing Information From a Specific Catalog

CATALOG Operand

You can list all of the information from a specific catalog at your terminal, or write it to a data set. To list all of the information from a specific catalog at your terminal, use the CATALOG operand and enclose the catalog name in parentheses.

Example

To list all information in catalog USERCAT1 at your terminal, enter after the READY mode message:

```
LISTCAT CATALOG('catalog.usercat1')
```

You then see something like the following:

```
⋮
USER1.NEW.DATA
USER1.PROGRAM.TEXT
USER3.PROG1.COBOL
USERID.PANEL.TEXT
⋮
PREFIX.DATA.DATA
PREFIX.LOG.MISC
⋮
```

CATALOG, OUTFILE Operands

To write all information in a specific catalog to a data set, enter the LISTCAT command and specify:

- The CATALOG operand and enclose the catalog name in parentheses.
- The OUTFILE (abbreviated OFILE) operand
  - The OUTFILE operand specifies the ddname of the DD statement or the name specified with the FILE operand of the ALLOCATE command, which defines the data set to which the system writes the list of catalog entries. To find out which DD statement defines the data set you want to write to, use the LISTALC command. For more information on how to do this, see “Displaying Ddnames and Data Set Disposition” on page 64.
  - When using the OUTFILE operand, enter the ddname or file name immediately after the operand, and enclose it in parentheses.

Example

To write all of the information in catalog USECAT1 to data set CATLIST1.DATA, where CATLIST.DATA is defined by DD statement CATDD, enter:

```
LISTCAT CATALOG('catalog.usecat1') OUTFILE(catdd)
```

If USECAT2 is a password-protected catalog, and the password is VGTBLS, enter a slash (/) and the catalog's password immediately following the catalog name field.

Example

To list all of the information in password-protected catalog USECAT2 at your terminal, enter:

```
LISTCAT CATALOG('catalog.usecat2'/vgtbls)
```

Listing Information by Data Set Qualifier

LEVEL Operand

To list information from a catalog associated with a particular qualifier, use the LEVEL operand and enclose the qualifier(s) in parentheses.

Example

To list information about all of the data sets in a catalog that have USER15 as the first qualifier, enter:

```
LISTCAT LEVEL(user15)
```

You might then see:

```
NONVSAM ------- USER15.FIRST.DATA
     IN-CAT --- CATALOG.USERCAT1
NONVSAM ------- USER15.PROJ2.DATA
     IN-CAT --- CATALOG.USERCAT1
```

To list information about all of the data sets in a catalog that have USER15 as the first qualifier and PROJ2 as the second qualifier, enter:

```
LISTCAT LEVEL(user15.proj2)
```

You might then see:

```
NONVSAM ------- USER15.PROJ2.DATA
     IN-CAT --- CATALOG.USERCAT1
```

Listing Alias Entries in a Catalog

ALIAS Operand

Catalogs can contain aliases, or alternate names for data sets. You can list all the aliases in a catalog by specifying the ALIAS operand. You need not specify any other information with this operand.

Example
To list all the alias entries in a catalog, enter: `LISTCAT ALIAS` If no aliases exist, you see the READY mode message.

Listing Data Set Information By Creation and Expiration Dates

CREATION Operand

You can list information about data sets that were created on or before a specified number of days from today by using the CREATION operand followed immediately by a decimal number enclosed in parentheses.

Example
To list information about data sets that were created ten days ago or before, enter: `LISTCAT CREATION(10)`

EXPIRATION Operand

You can list information about data sets that will expire on or before a specified number of days from today by using the EXPIRATION operand followed immediately by a decimal number enclosed in parentheses.

Example
To list the entries that will expire within the next eight days, enter: `LISTCAT EXPIRATION(8)`

Listing Specific Data Set Information in a Catalog

ALL Operand

Use the ALL operand to list each data set's:

- Name
- Owner identification
- Creation date
- Expiration date
- Entry type
- Volume serial number
- Device type.

To list the entries that will expire within the next eight days, enter:

```
LISTCAT EXPIRATION(8)
```

Example

To list each data set's name, owner identification, creation date, expiration date, entry type, volume serial number(s), and device type(s) in a catalog, enter:

```
LISTCAT ALL
```

You then see something like the following for each of your data sets:

```
NONVSAM ------- PREFIX.CLIST
  IN-CAT --- CATALOG.USERCAT1
  HISTORY
    OWNER-IDENT ------ (NULL)    CREATION -----1987.275
    RELEASE ----------------2    EXPIRATION----0000.000
  VOLUMES
    VOLSER ------------TSO001    DEVTYPE----X'3010200E'
  ASSOCIATIONS---------(NULL)
```

HISTORY Operand

To limit the amount of information you see, use the HISTORY operand to list only the name of each data set, the owner identification, creation date, and expiration date.

Example

To list only each data set's name, owner identification, creation date, and expiration date, enter:

```
LISTCAT HISTORY
```

You then see something like the following for each of your data sets:

```
NONVSAM ------- PREFIX.CLIST
  IN-CAT --- CATALOG.USERCAT1
  HISTORY
    OWNER-IDENT ------ (NULL)    CREATION -----1987.275
    RELEASE ----------------2    EXPIRATION----0000.000
```

For more information about the LISTCAT command, see *z/OS TSO/E Command Reference*. Additional information may be found in *z/OS DFSMS Access Method Services Commands*.

Listing Data Set Attributes - LISTDS Command

To display the attributes of specific data sets at your terminal, use the LISTDS command. Each data set you specify must be currently allocated or available from the system catalog, and must reside on a currently-active volume.

Using the LISTDS command, you can display:

- The volume identifier (VOLID) of the volume the data set resides on
- The logical record length (LRECL), the block size (BLKSIZE), and for non-VSAM data sets, the record format (RECFM) of the data set
- The data set organization (DSORG). For an explanation of the possible values for DSORG and their meanings, see “Listing the History of a Data Set” on page 65
- The date the data set was created
- The date the data set will expire
- For non-VSAM data sets, whether the data set is protected
- File name and disposition
- The members in a partitioned data set, including any aliases

- Non-VSAM data set control blocks (DSCBs)
- The information from the data sets in a particular user catalog
- A list of information about data sets with a common high-level qualifier

You can specify a list of data set names when using any of the operands of the LISTDS command.

Listing Data Set RECFM, LRECL, BLKSIZE, DSORG, and VOLID

You can list the RECFM, LRECL, BLKSIZE, DSORG, and VOLID of a specific data set by using the LISTDS command followed by a data set name.

Example

To list the attributes of data set PREFIX.PARTS.DATA, enter:

```
LISTDS parts.data
```

You then see something like:

```
PREFIX.PARTS.DATA
--RECFM-LRECL-BLKSIZE-DSORG
  VB    255   5100    PO
--VOLUMES--
  TSO001
```

You can also list the attributes of a number of specific data sets by entering the LISTDS command followed by the data set names enclosed in parentheses and separated by commas or blanks. Or you can use an asterisk (*) in place of any qualifier except the first. When you specify an asterisk, the system searches for all cataloged data sets whose names contain the qualifiers you specified. For instance, if you want to list the attributes of all the data sets starting with PREFIX.TEST, specify **LISTDS TEST.*** or **LISTDS 'PREFIX.TEST.*'**.

Example

To list the attributes of your data sets PARTS.DATA, ART.TEXT, and JONES.CNTL, enter:

```
LISTDS (parts.data,art.text,jones.cntl)
```

You then see the attributes for the three data sets.

Listing Data Sets' Creation and Expiration Dates

HISTORY Operand

By using the HISTORY operand, you can list in addition to the previous attributes:

- The date a data set was created
- The date the data set expires
- Whether the data set is password-protected (for non-VSAM data sets)

Example

To list the creation and expiration dates, and whether non-VSAM data set PARTS.DATA is password-protected, enter:

```
LISTDS parts.data HISTORY
```

You then see the following:

```
PREFIX.PARTS.DATA
--RECFM-LRECL-BLKSIZE-DSORG-CREATED---EXPIRES---SECURITY
  VB    255   5100    PO    1987.291   00.000   RACF
--VOLUMES--
  TSO001
```

Listing Data Sets' Associated Ddnames and Dispositions

STATUS Operand

You can list the ddname currently associated with a data set, and that data set's disposition by specifying the data set name followed by the STATUS operand.

Example

To list the ddname currently associated with data set PARTS.DATA, and its disposition, enter:

```
LISTDS parts.data STATUS
```

You then see something like the following:

```
PREFIX.PARTS.DATA
--RECFM-LRECL-BLKSIZE-DSORG--DDNAME---DISP
  VB    255   5100    PO     SYS00037 KEEP
--VOLUMES--
  TSO001
```

Listing the Members of a Partitioned Data Set

MEMBERS Operand

You can list all of the members in a partitioned data set, including any aliases, by specifying the data set name followed by the MEMBERS operand.

Example

To list all of the members in partitioned data set PARTS.DATA, including any aliases, enter:

```
LISTDS parts.data MEMBERS
```

You then see something like the following:

```
PREFIX.PARTS.DATA
--RECFM-LRECL-BLKSIZE-DSORG
  VB    255   5100    PO
--VOLUMES--
  TSO001
--MEMBERS--
  PART1
  PART2
  PART3
```

Listing the DSCB for a Non-VSAM Data Set

LABEL Operand

You can list the data set control block (DSCB) associated with a non-VSAM data set by specifying the data set name followed by the LABEL operand.

This operand is only applicable to non-VSAM data sets on direct access devices.

Example

To list the DSCB associated with non-VSAM data set PARTS.DATA, enter:

```
LISTDS parts.data LABEL
```

You then see in hexadecimal notation something like the following:

```
PREFIX.PARTS.DATA
--RECFM-LRECL-BLKSIZE-DSORG--DDNAME---DISP
  VB    255   5100    PO     SYS00037 KEEP
--VOLUMES--
  TSO001
--FORMAT 1 DSCB--
F1 E3D4D6F0F1F0 0001 540124 04 00 D2C2D4D6E1E2F240404040
570005000000 0200 50 13EC 00FF 00 80000002 000D14 6B80
010000E1000300E10 0100520800520008 01020340E03440E 03780
--FORMAT 3 DSCB--
03030303 010303690D03690E 000000000000000 000000000000000
0000000000000000 F3 0000000000000000 0000000000000000
0000000000000000 0000000000000000 0000000000000000
0000000000000000 0000000000000000 0000000000000000
0000000000000000000 000000000
```

Listing Information for Commonly Owned Data Sets

LEVEL Operand

You can list information for data sets that have a common high-level qualifier by specifying the high-level qualifier followed by the LEVEL operand. The data sets could be owned by a particular user or could be associated with a particular project that includes many users.

Example

To list all the data sets whose high-level qualifier is PROJ3, that is, all data sets associated with the prefix PROJ3, enter:

```
LISTDS 'proj3' LEVEL
```

You then see something like the following for each data set:

```
PROJ3.AM.TEXT
--RECFM-LRECL-BLKSIZE-DSORG
  VB    84    3360    PO
--VOLUMES--
  TSO002
```

When you specify the LEVEL operand, the data set name cannot contain an asterisk.

For more information on the LISTDS command, see *z/OS TSO/E Command Reference*.

Using ISPF/PDF to List Data Set Information

In addition to typing a line mode TSO/E command to list information about data sets, you can enter information on panels in ISPF/PDF to list similar information.

To list information about data sets, select the UTILITIES option (option 3) on the ISPF/PDF Primary Option Menu.

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 3
                                                              USERID   - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME     - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS  - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

From the Utilities Selection Menu, select the DSLIST option (option 4).

```
--------------------------  UTILITY SELECTION MENU  ----------------------------
OPTION ===> 4

    1  LIBRARY     - Compress or print data set.  Print index listing.
                     Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                     display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                     Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

You then see a panel that lets you specify what type of list you want and shows you what actions you can perform on the resulting list.

```
----------------------------  DATA SET LIST UTILITY  -------------------------
OPTION ===>

  blank - Display data set list *             P  - Print data set list
  V     - Display VTOC information only       PV - Print VTOC information only

Enter one or both of the parameters below:
  DSNAME LEVEL  ===>  PREFIX
  VOLUME        ===>


  INITIAL DISPLAY VIEW   ===>  VOLUME  (VOLUME,SPACE,ATTRIB,TOTAL)
  CONFIRM DELETE REQUEST ===>  YES    (YES or NO)

* The following line commands will be available when the list is displayed

B - Browse data set        C - Catalog data set       F - Free unused space
E - Edit data set          U - Uncatalog data set     = - Repeat last command
D - Delete data set        P - Print entire data set
R - Rename data set        X - Print index listing
I - Data set information   M - Display member list
S - Information (short)    Z - Compress data set      TSO command or CLIST
```

The list of data sets for PREFIX with a VOLUME format might look something like the following panel. The data sets listed at the beginning of the list are data sets for system use.

```
DSLIST - DATA SETS BEGINNING WITH PREFIX --------------------------  ROW 1 OF 9
COMMAND ===>                                                 SCROLL ===> PAGE
COMMAND      NAME                                  MESSAGE            VOLUME
------------------------------------------------------------------------------
         PREFIX.$10$09$5                                               MAR87B
         PREFIX.$15$38$5                                               OCT87A
         PREFIX.#OXYCALC.CDATA                                         MIGRAT
         PREFIX.#OXYCALC.CFORM                                         MIGRAT
         PREFIX.#OXYCALC.REPORT                                        TSO007
         PREFIX.#OXYCALC.SAVE                                          TSO005
         PREFIX.ABDRVR7.ISPPLIB                                        FEB87B
         PREFIX.MEMO.TEXT                                              NOV87C
         PREFIX.TEST.DATA                                              NOV87C
********************************** END OF DATA SET LIST **********************
```

For more information about using ISPF/PDF to list information about data sets, see *z/OS ISPF User's Guide Vol I*.

Chapter 7. Editing Data Sets

When you allocate a new data set, the data set is empty until you put data into it. One way to put data into a data set is by editing the data set. Editing also allows you to modify the contents of a data set that contains data.

To edit data sets in TSO/E, use:

- The EDIT command - a line mode editor
- The EDIT option of ISPF/PDF - a full-screen editor

Editing Data Sets with the EDIT Command

The line mode TSO/E editor has two modes of operation, INPUT mode and EDIT mode. If the data set is new, the editor goes into INPUT mode and you see a line number for a line of input. If the data set already exists and contains data, the editor goes into EDIT mode.

You can change from EDIT mode to INPUT mode by typing the word `input` or by entering a null line. You can change from INPUT mode to EDIT mode by pressing the Enter key after a blank line of input.

The EDIT command has many subcommands, one of which is LIST. When you are in EDIT mode, you can specify the subcommand LIST to display the contents of the data set.

© Copyright IBM Corp. 1988, 2024

Example

To allocate and edit a new data set PREFIX.REPORT.TEXT with default attributes, enter:

```
 EDIT report.text
```

You then see:

```
DATA SET OR MEMBER NOT FOUND, ASSUMED TO BE NEW
INPUT
00010
```

The INPUT subcommand indicates that you are in input mode. You can then type a line of data and press the Enter key to display the next available line.

```
INPUT
00010 This is the first line of data.
00020
```

To get out of INPUT mode, press the Enter key without typing information on the available line. You then go into EDIT mode:

```
INPUT
00010 This is the first line of data.
00020
EDIT
```

While in EDIT mode, you can type an EDIT subcommand, such as END, LIST, SAVE, or RUN. To read about these and other subcommands of EDIT, see Appendix B, “Using Line Mode Edit,” on page 195.

To simply save the information and end the edit session, enter the SAVE subcommand followed by the END subcommand.

```
 INPUT
 00010 This is the first line of data.
 00020
 EDIT
save
 EDIT
end
 READY
```

For more information on the EDIT command, see Appendix B, “Using Line Mode Edit,” on page 195.

Using the EDIT Option of ISPF/PDF

To use a full-screen editor to edit a data set, select the EDIT option (option 2) from the ISPF/PDF Primary Option Menu.

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 2
                                                              USERID   - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME     - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS  - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

Then specify the data set name in either the ISPF LIBRARY fields or in the OTHER PARTITIONED OR SEQUENTIAL DATA SET field. Refer to “Specifying a Data Set Name” on page 54 for information about the ways to specify a data set name in these fields.

```
------------------------------  EDIT - ENTRY PANEL  --------------------------
COMMAND ===>

ISPF LIBRARY:
   PROJECT ===> prefix
   GROUP   ===> memo     ===>            ===>            ===>
   TYPE    ===> text
   MEMBER  ===>                       (Blank for member selection list)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   DATA SET NAME     ===>
   VOLUME SERIAL     ===>            (If not cataloged)

   DATA SET PASSWORD ===>            (If password protected)

   PROFILE NAME      ===>            (Blank defaults to data set type)

   INITIAL MACRO     ===>            LOCK       ===> YES    (YES, NO or NEVER)

   FORMAT NAME       ===>            MIXED MODE ===> NO     (YES or NO)
```

If the data set is partitioned, you can either specify the member name or you can omit the member name to display a list of members. If you display a list of members, select a member by entering S next to the member name on the extreme left.

```
LIBRARY - PREFIX.MEMO.TEXT ---------------------------------- ROW 00001 OF 00008
COMMAND ===>                                                   SCROLL ===> PAGE
   NAME     RENAME    VV.MM   CREATED       CHANGED      SIZE   INIT   MOD  ID
  APR13               01.02  87/04/13   87/05/21 09.55    44     39     4 YOURID
  AUG01               01.00  87/08/01   87/08/01 10.01    24     24     0 YOURID
  FEB27               01.01  87/02/27   87/03/07 14.52    12     11     2 YOURID
  JAN10               01.00  87/01/10   87/01/10 17:07    21     21     0 YOURID
  JUL01               01.01  87/07/07   87/07/07 12.52    10     10     0 YOURID
s JUL14               01.00  87/07/14   87/07/14 16.15    20     20     0 YOURID
  JUN04               01.00  87/06/04   87/06/04 11.23    85     85     0 YOURID
  JUN18               01.04  87/06/18   87/06/27 08.43    34     36     6 YOURID
  **END**
```

The full-screen edit display panel allows you to alter the contents of a data set on a full screen by typing over information, deleting information, and adding or changing lines of information.

```
EDIT ---- PREFIX.MEMO.TEXT(JUL14)  ---------------------------- COLUMNS 001 072
COMMAND ===>                                                  SCROLL ===> HALF
****** ************************ TOP OF DATA ***********************************
000001 To:  Accounting Department
000002      ABC Company
000003      120 Kings Highway
000004      Smithfield, N.J.
000005
000006 Subject:  Raises
000007
000008 This is to inform you that there will be an across-the-board
000009 raise of 6% for all employees due to the spectacular profits
000010 the ABC Company made this past year.
000011
000012 Please include this increase in the next pay check.
****** ************************** BOTTOM OF DATA ******************************
```

To save information entered on the edit display panel, enter `save` on the COMMAND line. To save and end the edit session, press the END PF key. To cancel information entered but not saved during the edit session, enter `cancel` on the COMMAND line. For more information about editing data sets in ISPF/PDF, see *z/OS ISPF User's Guide Vol I* and *z/OS ISPF Edit and Edit Macros*.

Chapter 8. Renaming Data Sets

Sometimes it is necessary to rename a data set or a member of a data set when, for clarification, you need to revise the data set name.

To rename data sets and members of data sets:

- Use the RENAME command
- Use the UTILITIES option of ISPF/PDF

Renaming Data Sets with the RENAME Command

Use the RENAME command to:

- Change the name of a single volume, non-VSAM cataloged, non-SMS-managed data set.
- Change the name of a single or multivolume, non-VSAM cataloged, SMS-managed data set.
- Change the name of a member of a cataloged partitioned data set
- Create an alias for a member of a cataloged partitioned data set.

To change the name of a VSAM data set, you can use the Access Method Services ALTER command. For more information about the ALTER command, see *z/OS DFSMS Access Method Services Commands*.

When you rename a password protected data set, the data set does not retain the password. If you want to password protect the renamed data set, you must use the PROTECT command to assign a password to the data set before you access it.

Do not use the RENAME command to create an alias for a load module created by the linkage editor.

Changing a Data Set's Name

To change the name of a data set, specify on the RENAME command:

- The data set name you want to change
- The data set's new name

Example

To change the name of data set PREFIX.TEST.DATA to PREFIX.PARTS.DATA, enter:

```
RENAME test.data parts.data
```

Renaming a Group of Data Sets

To rename a group of data sets that have at least one qualifier in common, issue the RENAME command and specify asterisks in place of the qualifiers that vary in both the new and old data set names. Remember that you are renaming all data sets with the specified common qualifier.

© Copyright IBM Corp. 1988, 2024

Example

Given the following data sets:

```
PREFIX.TEST.DATA
PREFIX.PARTS.DATA
PREFIX.OLD.DATA
PREFIX.WORK.DATA
```

To change the third qualifier in all the data sets to TEXT, enter:

```
RENAME *.data *.text
```

Renaming a Member

To change the name of a member of a partitioned data set, issue the RENAME command and:

- Specify the data set name and the old member name. Enclose the old member name in parentheses.
- Optionally, specify the data set name again.
- Enclose the new member name in parentheses.

Example

To rename MEMBER1 of data set PREFIX.TEST.DATA to MEMBER2, enter:

```
RENAME test.data(member1) test.data(member2)
```

or

```
RENAME test.data(member1) (member2)
```

Creating an Alias Name for a Member

ALIAS Operand

To create an alias name for a member of a partitioned data set, issue the RENAME command and:

- Specify the data set name and the member name. Enclose the member name in parentheses.
- Optionally, specify the data set name again.
- Enclose the member's alias in parentheses.
- Specify the ALIAS operand.

Example

To create AUXMEM as an alias for MEMBER1 of data set PREFIX.TEST.DATA, enter:

```
RENAME test.data(member1) test.data(auxmem) ALIAS
```

or

```
RENAME test.data(member1) (auxmem) ALIAS
```

For more information on the RENAME command, see *z/OS TSO/E Command Reference*.

Renaming Data Sets with the UTILITIES Option of ISPF/PDF

You can rename entire data sets or individual members of a partitioned data set with the ISPF/PDF UTILITIES option.

Renaming an Entire Data Set

To rename a sequential data set or a partitioned data set's first three qualifiers, select the UTILITIES option (option 3) from the ISPF/PDF Primary Option Menu.

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 3
                                                          USERID   - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME     - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS  - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

You then see the Utilities Selection Menu. To rename an entire data set, select the DATASET option (option 2).

```
-------------------------  UTILITY SELECTION MENU  --------------------------
OPTION ===> 2

    1  LIBRARY     - Compress or print data set.  Print index listing.
                     Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                     display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                     Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

On the next panel, the Data Set Utility panel, specify the name of the data set you want to change in either the ISPF LIBRARY fields or in the OTHER PARTITIONED OR SEQUENTIAL DATA SET field. Refer to “Specifying a Data Set Name” on page 54 for information about specifying the data set name in these fields. Then type an R on the OPTION line.

```
------------------------------  DATA SET UTILITY  ----------------------------
OPTION ===> r

    A - Allocate new data set                 C - Catalog data set
    R - Rename entire data set                U - Uncatalog data set
    D - Delete entire data set                S - Data set information (short)
    blank - Data set information

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  test
    TYPE    ===>  data

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>          (If not cataloged, required for option "C")

DATA SET PASSWORD ===>          (If password protected)
```

You see another panel, the Rename Data Set panel, on which you specify the new name for the data set. When you press the Enter key on this panel, the old data set name no longer exists.

```
-------------------------------  RENAME DATA SET  ----------------------------
COMMAND ===>

DATA SET NAME: PREFIX.TEST.DATA
VOLUME:        TSO014

ENTER NEW NAME BELOW:      (The data set will be recataloged.)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  newtest
    TYPE    ===>  data

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
```

For more information about renaming data sets, see *z/OS ISPF User's Guide Vol I*.

Renaming a Data Set Member

To rename a member of a partitioned data set, select the UTILITIES option from the ISPF/PDF Primary Option Menu. Instead of selecting the DATASET option from the Utility Selection Menu, however, select the LIBRARY option (option 1).

```
--------------------------  UTILITY SELECTION MENU  ---------------------------
OPTION ===> 1

    1  LIBRARY     - Compress or print data set.  Print index listing.
                     Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                     display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                     Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

You then see the Library Utility panel. To rename one member of a partitioned data set, specify the data set name including the member's current name in the ISPF LIBRARY fields. Specify the member's new name in the NEWNAME field, and type R on the OPTION line.

```
------------------------------  LIBRARY UTILITY  ------------------------------
OPTION ===> R

    blank - Display member list              B - Browse member
    C - Compress data set                    P - Print member
    X - Print index listing                  R - Rename member
    L - Print entire data set                D - Delete member
    I - Data set information                 S - Data set information (short)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  memo      ===>             ===>            ===>
    TYPE    ===>  text
    MEMBER  ===>  aug01       (If option "P", "R", "D", or "B" selected)
    NEWNAME ===>  oct10       (If option "R" selected)


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>              (If not cataloged)

DATA SET PASSWORD ===>              (If password protected)
```

Renaming More than One Member

To rename more than one member in a partitioned data set, specify the first three qualifiers of the partitioned data set and leave the OPTION line blank.

```
------------------------------  LIBRARY UTILITY  ------------------------------
OPTION ===>

    blank - Display member list              B - Browse member
    C - Compress data set                    P - Print member
    X - Print index listing                  R - Rename member
    L - Print entire data set                D - Delete member
    I - Data set information                 S - Data set information (short)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  memo      ===>             ===>            ===>
    TYPE    ===>  text
    MEMBER  ===>                      (If option "P", "R", "D", or "B" selected)
    NEWNAME ===>                      (If option "R" selected)


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>              (If not cataloged)

DATA SET PASSWORD ===>              (If password protected)
```

You then see a list of members. To rename a member, type R next to the member name and type the new member name in the RENAME column. When you press the Enter key, the word RENAMED appears in the RENAME column.

```
LIBRARY - PREFIX.MEMO.TEXT -------------------------------- ROW 00001 OF 00008
COMMAND ===>                                                  SCROLL ===> PAGE
   NAME     RENAME    VV.MM   CREATED       CHANGED      SIZE   INIT   MOD  ID
  APR13                01.02  87/04/13   87/05/21 09.55    44     39     4 YOURID
r AUG01    OCT10       01.00  87/08/01   87/08/01 10.01    24     24     0 YOURID
  FEB27                01.01  87/02/27   87/03/07 14.52    12     11     2 YOURID
  JAN10                01.00  87/01/10   87/01/10 17:07    21     21     0 YOURID
r JUL01    MAY02       01.01  87/07/07   87/07/07 12.52    10     10     0 YOURID
  JUL14                01.00  87/07/14   87/07/14 16.15    20     20     0 YOURID
  JUN04                01.00  87/06/04   87/06/04 11.23    85     85     0 YOURID
  JUN18                01.04  87/06/18   87/06/27 08.43    34     36     6 YOURID
  **END**
```

When you finish renaming members, press the END PF key to exit the panel. A message that says how many members were renamed appears in the top right hand corner of the screen. For more information about renaming data set members, see *z/OS ISPF User's Guide Vol I*.

Chapter 9. Copying Data Sets

Often when managing data sets, you need to copy information from one data set to another. You can copy parts of existing data sets to form new data sets or you can copy an entire data set and revise it.

There are several ways you can copy information from one data set to another:

- Use the SMCOPY command. Although SMCOPY is primarily intended for use under Session Manager, you can use it to copy some data set types when you are not logged on under Session Manager.
- Use the UTILITIES option of ISPF/PDF.

Copying Data Sets with the SMCOPY Command

Use the SMCOPY command to copy:

- A member of a partitioned data set into another member or into a sequential data set.

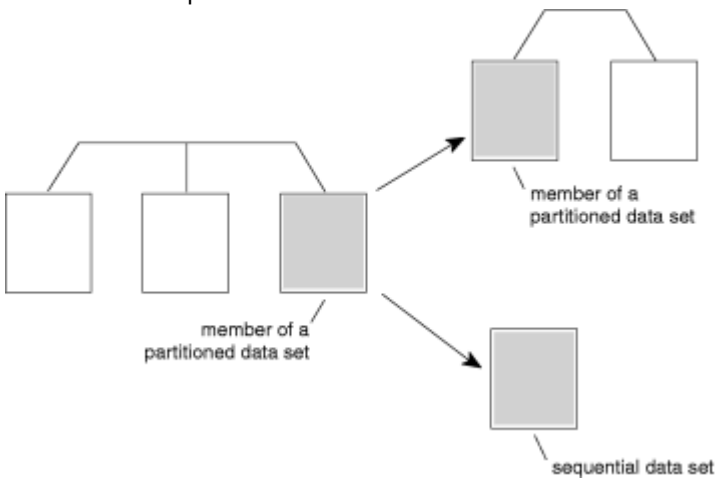


Figure 2. Copying a Member of a Partitioned Data Set

- A sequential data set into another sequential data set or into a member of a partitioned data set.

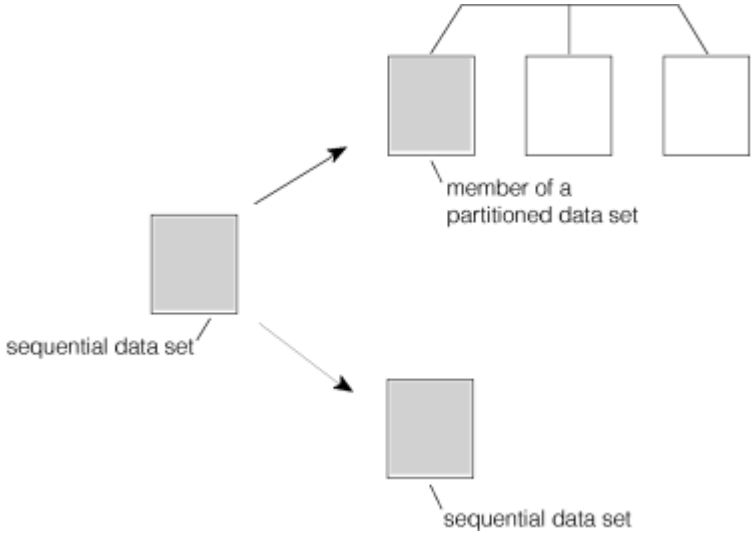


Figure 3. Copying a Sequential Data Set

The data sets or members you copy must have:

- A variable (V) or fixed (F) record format. (The records can be blocked or unblocked.)

© Copyright IBM Corp. 1988, 2024

- A logical record length of no more than 256 bytes.

To copy other types of data sets, such as load libraries, which have a record format of U, use ISPF/PDF.

Operands of SMCOPY

When using the SMCOPY command, specify the following operands:

- FROMDATASET(dsname) or FDS(dsname) to designate the originating sequential data set or member of a partitioned data set
- TODATASET(dsname) or TDS(dsname) to designate the destination sequential data set or member of a partitioned data set
- NOTRANS to prevent characters in the data set from changing to upper case and unprintable characters from changing to blanks

You can copy a data set to either a new or existing data set. When you copy to a new data set, the system allocates one for you, using the attributes of the data set being copied. When you copy to an existing data set, the system uses the attributes of the data set to which you are copying the data.

You can copy data sets that have different logical record lengths or block sizes. You cannot, however, copy data sets that have different record formats. For example, you cannot copy a data set with a record format of FB to a data set with a record format of VB.

Example

To copy member MEM1 of the partitioned data set PREFIX.PDS1.DATA into member MEM2 of PREFIX.PDS2.DATA, specify:

```
SMCOPY FDS(pds1.data(mem1)) TDS(pds2.data(mem2)) NOTRANS
```

To copy member MEM1 of the partitioned data set PREFIX.PDS1.DATA into the sequential data set PREFIX.SEQ1.DATA, specify:

```
SMCOPY FDS(pds1.data(mem1)) TDS(seq1.data) NOTRANS
```

Copying Part of a Data Set

You can copy specific records from one sequential data set or member to another. Using the SMCOPY command, specify:

- The FDS and TDS operands, as shown above.
- The LINE operand with the range of records to be copied enclosed in parentheses. Specify a starting record and an ending record.

Example

To copy the first five records of text from the member MEM1 of PREFIX.PDS1.DATA to MEM2 of PREFIX.PDS2.DATA, specify:

```
SMCOPY FDS(pds1.data(mem1)) LINE(1:5) TDS(pds2.data(mem2)) NOTRANS
```

You cannot use the LINE operand to copy specific line numbers within a numbered data set. The system treats the numbers you specify as offsets from the beginning of the data set.

For more information about the SMCOPY command, see Chapter 18, “Session Manager,” on page 163 and *z/OS TSO/E Command Reference*.

Using ISPF/PDF to Copy a Data Set

There are several ways to copy information using ISPF/PDF. You can use the UTILITIES option to copy one data set to another. You can also use an ISPF/PDF COPY command with line commands to copy parts of one data set to another. For more information about copying parts of data sets, see *z/OS ISPF User's Guide Vol I*.

Copying One Data Set to Another

Like the SMCOPY command, you can use the UTILITIES option to copy:

- A sequential data set to another sequential data set or to a member of a partitioned data set (See Figure 2 on page 87.), or
- A member of a partitioned data set to another member of a partitioned data set, or to a sequential data set. (See Figure 3 on page 87.)

In addition, you can copy:

- An entire partitioned data set to another partitioned data set.

To copy one data set to another, select the UTILITIES option (option 3) on the ISPF/PDF Primary Option Menu.

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 3
                                                          USERID    - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME      - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS   - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

You then see the Utility Selection Menu. To copy one data set to another, select the MOVE/COPY option (option 3).

```
-------------------------  UTILITY SELECTION MENU  ----------------------------
OPTION ===> 3

    1  LIBRARY     - Compress or print data set.  Print index listing.
                     Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                     display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                     Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

On the next panel, the Move/Copy Utility panel, specify the name of the data set **from** which you want to copy.

For information about specifying data set names, refer to "Specifying a Data Set Name" on page 54. You can specify a sequential data set, a member of a partitioned data set, or an entire partitioned data set by typing * in the MEMBER field. The following example shows a sequential data set.

```
------------------------------  MOVE/COPY UTILITY  ------------------------------
OPTION ===> c

   C - Copy data set or member(s)              CP - Copy and print
   M - Move data set or member(s)              MP - Move and print
   L - Copy and lock member(s)                 LP - Copy, lock, and print
   P - Promote data set or member(s)           PP - Promote and print

SPECIFY "FROM" DATA SET BELOW, THEN PRESS ENTER KEY

FROM ISPF LIBRARY:
   PROJECT ===> prefix
   GROUP   ===> report1
   TYPE    ===> text
   MEMBER  ===>                 (Blank or pattern for member selection list,
                                 '*' for all members)

FROM OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   DATA SET NAME  ===>
   VOLUME SERIAL  ===>          (If not cataloged)

DATA SET PASSWORD ===>          (If password protected)
```

On the following panel, specify the data set **to** which you want the copied information to go.

```
COPY --- FROM PREFIX.REPORT1.TEXT ----------------------------------------------
COMMAND ===>

SPECIFY "TO" DATA SET BELOW.

TO ISPF LIBRARY:
   PROJECT ===> prefix
   GROUP   ===> report2
   TYPE    ===> text
   MEMBER  ===>                 (Blank for member list, * for all members)

TO OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   DATA SET NAME  ===>
   VOLUME SERIAL  ===>          (If not cataloged)

DATA SET PASSWORD ===>          (If password protected)

"TO" DATA SET OPTIONS:
   IF PARTITIONED, REPLACE LIKE-NAMED MEMBERS ===> YES     (YES or NO)
   IF SEQUENTIAL, "TO" DATA SET DISPOSITION   ===> OLD     (OLD or MOD)
   SPECIFY PACK OPTION FOR "TO" DATA SET      ===>         (YES, NO or blank)
```

When you press the Enter key on the above panel, the entire data set or member is then copied over the other data set or member and the original information is lost. For more information about copying data sets, see *z/OS ISPF User's Guide Vol I*.

Chapter 10. Sending and Receiving Data Sets

In addition to sending and receiving messages, you can send and receive data sets using the TRANSMIT and RECEIVE commands.

- Use the TRANSMIT command to:
  - Send data sets to other users
  - Send a message with a data set to other users
  - Send a data set that is displayed on the terminal when it is received
- Use the RECEIVE command to:
  - Receive data sets from other users

A typical TRANSMIT command to transmit a data set is:

```
TRANSMIT node_id.user_id DATASET(dataset.name)
```

The TRANSMIT command does not support:

- Data sets with keys
- Data sets with user labels
- ISAM data sets
- VSAM data sets

For information about data sets used with TRANSMIT/RECEIVE, see "Using Nicknames and the Names Data Set" on page 30 and "Storing Transmitted Data in a Log" on page 34.

Sending a Data Set with the TRANSMIT Command

Using the TRANSMIT command, you can send the following to another user:

- A data set
- One or more members of a partitioned data set
- A message with a data set

Transmitting a Data Set

To transmit a data set or one member of a partitioned data set to another user, type the TRANSMIT command and specify:

- The destination node, followed immediately by a period and the destination user ID
- The DATASET operand with the data set name enclosed in parentheses

Example

To transmit data set PREFIX.SAMPLES.TEXT to user USER1 at node NODEID, enter:

```
TRANSMIT nodeid.user1 DATASET(samples.text)
```

To transmit member SAMP1 of data set PREFIX.SAMPLES.TEXT to USER1 at NODEID, enter:

```
TRANSMIT nodeid.user1 DATASET(samples.text(samp1))
```

Transmitting Selected Members of a Partitioned Data Set

© Copyright IBM Corp. 1988, 2024

MEMBERS Operand

To transmit selected members of a partitioned data set to another user, type the TRANSMIT command and specify:

- The destination node, followed immediately by a period and the destination user ID
- The DATASET operand with the data set name enclosed in parentheses
- The MEMBERS operand with the member names enclosed in parentheses. Within the parentheses, separate the member names with commas or blanks

Example

To send members ONE and TWO of data set PREFIX.SAMPLE.TEXT to user USER1 at node NODEID enter:

```
TRANSMIT nodeid.user1 DATASET(sample.text) MEMBERS(one,two)
```

Transmitting a Data Set and a Message

MESSAGE Operand

To transmit a data set with an accompanying message to another user, type the TRANSMIT command and specify:

- The destination node, followed immediately by a period and the destination user ID
- The DATASET operand with the data set name enclosed in parentheses
- The MESSAGE operand

The message is displayed either when the receiver logs on or when the receiver issues the RECEIVE command. The receiver sees the message before actually getting the data set, in case the sender wants to include special processing instructions in the message.

Example

To send data set PREFIX.SAMPLES.TEXT with an accompanying message to user USER11 at node NODEID, enter:

```
TRANSMIT nodeid.user11 DATASET(samples.text) MESSAGE
```

The system then prompts you to enter the message text in either full-screen mode or in line mode, depending on the default for the type of terminal you are using.

If you specify MESSAGE along with the default, TERMINAL, the system prompts you twice for the message text.

Transmitting a Data Set That Appears as a Message

MSGDATASET Operand

If the data set is a sequential data set or member of a partitioned data set with the following characteristics, it can be displayed on the screen just as a message is displayed:

- Record format (RECFM) of F or FB
- Record length (LRECL) of 80

To transmit a data set so that it appears on the receiver's screen when it is received, type the TRANSMIT command and specify:

- The destination node, followed immediately by a period and the destination user ID
- The MSGDATASET operand with the data set name enclosed in parentheses.

Example

To send data set member PREFIX.SAMPLES.TEXT(ONE) to USER3 at NODEID and have it display on the screen when it is received, enter:

```
TRANSMIT nodeid.user3 MSGDATASET(samples.text(one))
```

When USER3 issues the RECEIVE command, the data set member is displayed on the screen.

If the data set has been allocated to a file, you can use the MSGDDNAME or MGSFILE operands. For more information about the TRANSMIT command, see *z/OS TSO/E Command Reference*.

Note: The TRANSMIT command may work differently if your installation uses security labels and security checking. See "Security Considerations for Sending and Receiving Data Sets" on page 94 for more information.

Receiving Data Sets with the RECEIVE Command

When you issue the RECEIVE command and a data set was sent to you, unless the data set was sent as a message, you see something like the following:

```
Dataset A.DATASET.NAME from USER1 on NODEID
Enter restore parameters or 'DELETE' or 'END' +
```

You then have the following options:

- Receive the data set by pressing the Enter key. If the receive is successful, the new data set name is the same as the one transmitted except your prefix replaces the first qualifier.
- Rename the data set by entering under the message:

  ```
  DATASET(new.name)
  ```

- Delete the data set by entering:

  ```
  DELETE
  ```

- Postpone receiving the data set by entering:

  ```
  END
  ```

The RECEIVE command cannot, in general, reformat data sets. Be sure that you tell the system to write the transmitted data into a data set that has the same record format as the original data set. Be sure that the record length is compatible (equal for fixed-length records and equal or longer for variable-length records). Also, be sure to specify a block size that is compatible with both the record length and record format of the transmitted data. If the system detects a mismatch in block size, record length, or record format, the system terminates the RECEIVE command and issues the appropriate error messages.

Note: The RECEIVE command may work differently if your installation uses security labels and security checking. See "Security Considerations for Sending and Receiving Data Sets" on page 94 for more information.

Example

To receive data set EXAMPLES.TEXT transmitted to your user ID from USER8, enter:

```
RECEIVE
```

You then see:

```
Dataset A.EXAMPLES.TEXT from USER8 on NODEID
Enter restore parameters or 'DELETE' or 'END' +
```

To receive the data into a data set named PREFIX.EXAMPLES.TEXT, press the Enter key.

You then see:

```
Restore successful to dataset 'PREFIX.EXAMPLES.TEXT'
------------------------------------------------
No more files remain for the receive command to process.
```

If you receive a data set with the same name as one you already have, you see the message:

```
Dataset 'PREFIX.EXAMPLES.TEXT' already exists.  Reply 'R' to replace it.
```

If you want to replace it, enter R. If you don't want to replace it, press the Enter key to terminate the RECEIVE command. Then reissue the RECEIVE command and rename the data set.

To receive a data set accompanied by a message, enter RECEIVE with no operands, as in the preceding example.

When you receive a data set, the system prompts you for information to control how the data set is restored. You can accept the system defaults offered with the prompts, or supply additional operands with the RECEIVE command, as described in the *z/OS TSO/E Command Reference*.

Security Considerations for Sending and Receiving Data Sets

The TRANSMIT and RECEIVE commands may work differently depending on which security options are used on your system. For example, if your installation uses security labels and security checking, be aware of the following considerations:

- When you transmit a data set, the security label you are logged on with is associated with the transmitted data set.
- You can only receive data sets that you are authorized to receive based on the security label you are logged on with.
- If you cannot log on with a high enough security label to receive the data set, the system deletes the data set. However, your installation may use a JES installation exit to take some other action.
- If you have data sets to receive with security labels that are greater than the security label than you are logged on with, when you issue the RECEIVE command, RECEIVE does not display any information about the data sets. RECEIVE may display a system message that says you have no data sets to receive.
- If you log incoming messages when you use the RECEIVE command, be aware that a message is not logged if the default log data set is not at the same security label you are logged on with.

To receive data sets with a greater security label, you can log on with an appropriate security label, if your TSO/E user ID is authorized to do so.

Chapter 11. Printing Data Sets

There are many ways to print data sets. Your installation might have a number of system printers and a number of ways to access them. The following are two ways to print data sets:

- Use the PRINTDS command to send data sets to a system printer managed by the job entry subsystem (JES)
- Use the UTILITIES option of ISPF/PDF

Printing Data Sets with the PRINTDS Command

Use the PRINTDS command to:

- Print a sequential data set, one member of a partitioned data set, or an entire partitioned data set including the directory (DATASET operand)
- Print only the members or only the directory of a partitioned data set (MEMBERS, DIRECTORY operands)
- Print part of a data set (LINES operand)
- Specify the number of copies you want printed (COPIES operand)
- Specify an output class for JES to use (CLASS operand)
- Specify whether the output goes to a held output queue (HOLD/NOHOLD operands)
- Specify the name of an on-line data set to which the output goes (TODATASET operand)
- Control the maximum length of a printed line of output and specify what to do if the input line is longer than the output line (FOLD/TRUNCATE operands)
- Determine formatting characteristics of the data set (BIND, PAGELEN, BMARGIN, TMARGIN, and COLUMNS operands)
- Associate a specific group of print characteristics with a printer by referencing one or more output descriptors

Printing a Data Set

DATASET Operand

To print a sequential data set, a member of a partitioned data set, or an entire partitioned data set including the directory, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.

Example

To print sequential data set PREFIX.NEW.DATA, enter:
`PRINTDS DATASET(new.data)`
To print data set member PREFIX.MEMO.TEXT(OCT27), enter:
`PRINTDS DATASET(memo.text(oct27))`
To print all the members and the directory of the partitioned data set PREFIX.REPORT.TEXT, enter:
`PRINTDS DATASET(report.text)`

© Copyright IBM Corp. 1988, 2024

MEMBERS, DIRECTORY Operands

When printing a partitioned data set, you can print the members and not the directory by issuing the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The MEMBERS operand.

Example

To print all the members of partitioned data set PREFIX.REPORT.TEXT, enter:

```
PRINTDS DATASET(report.text) MEMBERS
```

To print only the directory of partitioned data set PREFIX.REPORT.TEXT, enter:

```
PRINTDS DATASET(report.text) DIRECTORY
```

Printing Part of a Data Set

LINES Operand

You can use the LINES operand to identify relative lines within a data set. To print a part of a data set, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The LINES operand immediately followed by the number of the first relative line, a colon, and the number of the last relative line of data you want printed. Enclose the numbers in parentheses.

If you specify only one number after the LINES operand, the data set is printed from the line specified to the end.

Example

To print lines 1 to 150 from data set PREFIX.NAMES.TEXT, enter:

```
PRINTDS DATASET(names.text) LINES(1:150)
```

Printing More than One Copy of a Data Set

COPIES Operand

To print more than one copy of a data set, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The COPIES operand followed by the number of copies from 1 to 255, enclosed in parentheses.

Example

To print five copies of PREFIX.REPORT.TEXT(YEAREND), enter:

```
PRINTDS DATASET(report.text(yearend)) COPIES(5)
```

The default is to print one copy.

Specifying a JES Output Class

CLASS Operand

To specify a JES output class other than the default class A or the default established by your installation, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The CLASS (or SYSOUT) operand followed by a single alphanumeric character enclosed in parentheses. The alphanumeric character must represent a valid output class at your installation.

Example

To specify JES output class J for data set PREFIX.REPORT.TEXT(YEAREND), enter:

```
PRINTDS DATASET(report.text(yearend)) CLASS(j)
```

Sending Data to a JES Hold Output Queue

HOLD/NOHOLD Operands

To hold data in a JES held output queue without printing it immediately, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The HOLD operand.

The default is NOHOLD, which indicates that the data is made available immediately for printing.

Example

To delay the printing indefinitely for data set PREFIX.TEST.DATA, enter:

```
PRINTDS DATASET(test.data) HOLD
```

Sending Formatted Data to Another Data Set

To send formatted data to an on-line data set from which you can view and edit the data before printing, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The TODATASET operand immediately followed by the name of a new or existing data set enclosed in parentheses.

If the data set named after the TODATASET operand has not yet been allocated, the PRINTDS command allocates the space for the new data set. If the data set named after the TODATASET operand already exists, new data from the PRINTDS command replaces the existing data.

Example

To format data from data set PREFIX.REPORT.TEXT(YEAREND) and send it to data set PREFIX.YEAREND.TEXT, enter:

```
PRINTDS DATASET(report.text(yearend)) TODATASET(yearend.text)
```

After viewing and editing PREFIX.YEAREND.TEXT, you can print the revised data by entering:

```
PRINTDS DATASET(yearend.text) notitle
```

Use the NOTITLE operand to suppress the default of TITLE, which causes a title to print at the top of each page. Because TITLE was the default for the first PRINTDS command in this example, you do not need a second default title with the second PRINTDS command. For more information about the TITLE/NOTITLE operands, see *z/OS TSO/E Command Reference*.

Controlling the Maximum Length of a Printed Line of Output

FOLD Operand

The FOLD operand gives the maximum length of a printed line of output and indicates that input longer than the specified length be continued on one or more following lines. To use the FOLD operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The FOLD operand followed by the column-width of the line enclosed in parentheses.

Example

To print data set PREFIX.TEST.DATA with lines no longer than 52 columns each, enter:

```
PRINTDS DATASET(test.data) FOLD(52)
```

TRUNCATE Operand

The TRUNCATE operand gives the maximum length of a printed line of output and indicates that input longer than the specified length be truncated. To use the TRUNCATE operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The TRUNCATE operand followed by the column-width of the line enclosed in parentheses.

Example

To print data set PREFIX.TEST.DATA with only the first 30 columns of each input line, enter:

```
PRINTDS DATASET(test.data) TRUNCATE(30)
```

Note: If neither the FOLD nor TRUNC operand is specified, the default line length is the length of the maximum input line.

Determining Formatting Characteristics for a Printed Data Set

BIND Operand

The BIND operand determines the left margin for printed output. To use the BIND operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The BIND (or LMARGIN) operand followed by the number of indenting columns enclosed in parentheses.

The BIND value must be from 0 to 255. If no BIND operand is specified, the default is BIND(0).

Example

To print data set PREFIX.TEXT.DATA and indent the left margin by 5 columns, enter:

```
PRINTDS DATASET(text.data) BIND(5)
```

PAGELEN Operand

The PAGELEN operand determines the number of printed lines on a page of output. The number of printed lines must be at least 6 and no more that 4095. To use the PAGELEN operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The PAGELEN operand followed by the number of lines per page enclosed in parentheses.

If you do not specify the PAGELEN operand, the default is 60 lines per page.

Example

To print data set PREFIX.TEST.DATA with 30 lines of output per page, enter:

```
PRINTDS DATASET(test.data) PAGELEN(30)
```

BMARGIN Operand

The BMARGIN operand determines the bottom margin for printed output in terms of blank lines. The number of blank lines for the bottom margin can be any number from 0 lines to 6 less than the PAGELEN value. In other words, the PAGELEN value minus the BMARGIN value and the TMARGIN value must be greater than or equal to 6. A printed page must contain at least 6 lines of data. To use the BMARGIN operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The BMARGIN operand followed by the number of blank lines to be left at the bottom of each page. Enclose the number of blank lines in parentheses.

If the BMARGIN operand is omitted, the default is BMARGIN(0).

Example

To print data set PREFIX.TEST.DATA with a bottom margin of 7 blank lines, enter:

```
PRINTDS DATASET(test.data) BMARGIN(7)
```

The resulting number of printed lines per page, using the default PAGELEN value of 60, is 53.

TMARGIN Operand

The TMARGIN operand determines the top margin for printed output in terms of blank lines at the top of each page. The number of blank lines for the top margin can be any number from 0 lines to 6 less than the PAGELEN value. In other words, the PAGELEN value minus the TMARGIN value and the BMARGIN value must be greater than or equal to 6. A printed page must contain at least 6 lines of data. To use the TMARGIN operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.

- The TMARGIN operand followed by the number of blank lines to be left at the top of each page. Enclose the number of blank lines in parentheses.

If the TMARGIN operand is omitted, the default is TMARGIN(0).

Example

To print data set PREFIX.TEST.DATA with a top margin of 8 blank lines, enter:

```
PRINTDS DATASET(test.data) TMARGIN(8)
```

To print the same data set with a top margin of 6, a bottom margin of 6, and 48 lines of printed data per page, enter:

```
PRINTDS DATASET(test.data) TMARGIN(6) BMARGIN(6) PAGELEN(60)
```

COLUMNS Operand

The COLUMNS operand identifies the columns of data to print from the input data set. You specify the columns in pairs with the first value being the starting column and the second value being the ending column. You can specify up to 32 column pairs. To use the COLUMNS operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The COLUMNS operand followed by the starting column value, a colon, and the ending column value for one or more pairs of columns, with the column pairs separated by commas. Enclose the column values in parentheses.

If an ending value is not specified, the last column of the input is used as the ending value.

Example

To print columns 6 through 10 and 15 through the last column, from data set PREFIX.TEXT.DATA, enter:

```
PRINTDS DATASET(text.data) COLUMNS(6:10,15)
```

The result is that information from column 6 through 10 of the input data set is printed in columns 1 through 5 of the output data set, and remaining information from columns 15 on appears starting in column 6 of the output data set.

Example

An eight-digit line number appears at the end of a 132-column data set named PREFIX.INPUT.DATA and occupies columns 125 to 132.

To print the line number at the beginning and follow it with the first 72 columns of information, enter:

```
PRINTDS DATASET(input.data) COLUMNS(125:132,1:72)
```

Associating a Group of Print Characteristics with a Printer

OUTDES Operand

The OUTDES operand specifies a name that associates output descriptors with a particular printer. The output descriptors are defined by OUTPUT JCL statements in the logon procedure and can contain formatting information such as fonts and whether pages from a 3800 printer should be burst into separate sheets. These output descriptors are then associated with a printer.

For example, the name "MEMPRINT" might be set up by an installation to associate memo formatting characteristics with a 6670 printer at a specific location. To send a memo to that printer, type `memprint` with the OUTDES operand:

```
OUTDES(memprint)
```

To use the OUTDES operand, issue the PRINTDS command with:

- The DATASET operand immediately followed by the data set name enclosed in parentheses.
- The OUTDES operand followed by the name of an output descriptor enclosed in parentheses. The name must be from 1 to 8 alphanumeric characters, the first of which must be alphabetic or one of the special characters #, $, or @. The output descriptor must have been previously defined in your logon procedure or by using the TSO/E OUTDES command (JES2 installations only).

You can specify a list of up to 128 output descriptors on the OUTDES operand, but typically you use only one.

Example

To print a report from data set PREFIX.REPORT.TEXT(YEAREND) on a printer that has been associated in your logon procedure with report formatting characteristics under the name REPORT1, enter:

```
PRINTDS DATASET(report.text(yearend)) OUTDES(report1)
```

Other PRINTDS Operands

In addition to the operands mentioned here, there are other PRINTDS operands. For example the CHARS operand allows you to select fonts. For information about the other operands and for more information about the PRINTDS command, see *z/OS TSO/E Command Reference*.

Using ISPF/PDF to Print a Data Set

To print an entire data set or to print one or more members of a partitioned data set, you can use the UTILITIES option of ISPF/PDF.

Before you can use ISPF/PDF to print a data set, you must:

- Define a job statement in ISPF/PDF. A job statement identifies you as the originator of a job you send to the printer.
- Establish a print and delete process option for the ISPF/PDF LIST data set. An ISPF/PDF LIST data set collects the data you want printed during your terminal session. When you exit ISPF/PDF, if you specified a print and delete process option for the LIST data set, the entire LIST data set is sent to the system printer.

Defining a Job Statement and LIST Default Process Option

A job statement in ISPF/PDF is written in job control language (JCL) and can contain your user ID, name, an account number, and the class and priority assigned to the job.

To define a job statement using ISPF/PDF and set a default process option for the LIST data set, select the ISPF PARMS option (option 0).

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 0
                                                              USERID   - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME     - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS  - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

On the next panel you see, select the LOG/LIST option (option 2).

```
---------------------------  ISPF PARAMETER OPTIONS  ---------------------------
OPTION ===> 2

    1  TERMINAL      - Specify terminal characteristics
    2  LOG/LIST      - Specify ISPF log and list defaults
    3  PF KEYS       - Specify PF keys for 3278 terminal with 24 PF keys
    4  DISPLAY       - Specify screen display characteristics
    5  LIST          - Specify list data set characteristics
    6  GRAPHIC       - Specify GDDM graphic print parameters
    7  ENVIRON       - Specify ENVIRON command settings
```

At the top right of the Log and List Defaults panel, type in the Process option field:

- PD - To print and delete the LIST data set
- D - To delete the LIST data set
- K - To keep the LIST data set
- KN - To allocate and keep a new LIST data set

At the bottom of the Log and List Defaults panel under the JOB STATEMENT INFORMATION field, type a job statement that looks something like the following:

```
-------------------------  LOG and LIST DEFAULTS  ----------------------------
COMMAND ===>

LOG DATA SET DEFAULT OPTIONS             LIST DATA SET DEFAULT OPTIONS
----------------------------             ----------------------------
Process option    ===> D                Process option    ===> PD
SYSOUT class      ===> A                 SYSOUT class      ===> A
Local printer ID  ===>                   Local printer ID  ===>
Lines per page    ===> 60                Lines per page    ===> 60
Primary pages     ===> 10                Primary pages     ===> 100
Secondary pages   ===> 10                Secondary pages   ===> 200

VALID PROCESS OPTIONS:
 PD - Print data set and delete          D - Delete data set (without printing)
 K  - Keep data set (append subsequent information to same data set)
 KN - Keep data set and allocate new data set

JOB STATEMENT INFORMATION:        (Required for system printer)
  ===>//YOURIDA  JOB  (ACCOUNT4),'YOUR NAME',
  ===>// CLASS=5,MSGCLASS=W,NOTIFY=YOURID
  ===>//* COMMENT
  ===>//* COMMENT
```

For more information about how to write a job statement, see *z/OS MVS JCL User's Guide*.

Printing a Data Set

To print an entire data set using ISPF/PDF or to print a member of a partitioned data set, select the UTILITIES option (option 3) from the ISPF/PDF Primary Option Menu.

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 3
                                                              USERID   - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME     - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS  - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

From the Utilities Selection Menu, select the LIBRARY option (option 1).

```
--------------------------  UTILITY SELECTION MENU  ---------------------------
OPTION ===> 1

    1  LIBRARY     - Compress or print data set.  Print index listing.
                     Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                     display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                     Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

Printing an Entire Data Set

To print an entire data set, type L on the OPTION line and specify the data set name in the ISPF LIBRARY fields or in the OTHER PARTITIONED OR SEQUENTIAL DATA SET field. Refer to "Specifying a Data Set Name" on page 54 for information about specifying the data set name in these fields. When you press the Enter key, the data set is sent to the LIST data set to await your exit from ISPF/PDF.

```
------------------------------  LIBRARY UTILITY  ----------------------------
OPTION ===> L

    blank - Display member list              B - Browse member
    C - Compress data set                    P - Print member
    X - Print index listing                  R - Rename member
    L - Print entire data set                D - Delete member
    I - Data set information                 S - Data set information (short)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  test      ===>            ===>            ===>
    TYPE    ===>  data
    MEMBER  ===>                    (If option "P", "R", "D", or "B" selected)
    NEWNAME ===>                    (If option "R" selected)


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>              (If not cataloged)

DATA SET PASSWORD ===>              (If password protected)
```

Printing Data Set Members

To print one member of a data set, type P on the OPTION line of the LIBRARY UTILITY panel and specify the three data set qualifiers plus the member name in the ISPF LIBRARY fields. When you press the Enter key, the member is sent to the LIST data set to await your exit from ISPF/PDF.

```
------------------------------  LIBRARY UTILITY  ----------------------------
OPTION ===> p

    blank - Display member list              B - Browse member
    C - Compress data set                    P - Print member
    X - Print index listing                  R - Rename member
    L - Print entire data set                D - Delete member
    I - Data set information                 S - Data set information (short)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  memo      ===>            ===>            ===>
    TYPE    ===>  text
    MEMBER  ===>  feb27       (If option "P", "R", "D", or "B" selected)
    NEWNAME ===>                    (If option "R" selected)


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>              (If not cataloged)

DATA SET PASSWORD ===>              (If password protected)
```

To print more than one member of a data set, you must display a member list from which you can select members. To display the member list, leave the OPTION line blank on the LIBRARY UTILITY panel. Then specify the three data set qualifiers in the ISPF LIBRARY field and leave the MEMBER field blank.

```
------------------------------  LIBRARY UTILITY  -----------------------------
OPTION ===>

    blank - Display member list              B - Browse member
    C - Compress data set                    P - Print member
    X - Print index listing                  R - Rename member
    L - Print entire data set                D - Delete member
    I - Data set information                 S - Data set information (short)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  memo      ===>            ===>            ===>
    TYPE    ===>  text
    MEMBER  ===>                    (If option "P", "R", "D", or "B" selected)
    NEWNAME ===>                    (If option "R" selected)


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>              (If not cataloged)

DATA SET PASSWORD ===>              (If password protected)
```

You then see a list of member names and information about each member. To print members, type p next to the members you want printed. (Notice that you can also browse, rename, and delete members from this option.) When you press the Enter key, the word '*PRINTED' appears next to the members you selected and the member is sent to the LIST data set. When you are finished printing members, press the END PF key (PF 3).

```
LIBRARY - PREFIX.MEMO.TEXT ---------------------------------- ROW 00001 OF 00008
COMMAND ===>                                                   SCROLL ===> PAGE
   NAME     RENAME    VV.MM   CREATED       CHANGED      SIZE   INIT   MOD  ID
  APR13               01.02  87/04/13   87/05/21 09.55    44     39     4 YOURID
p AUG01               01.00  87/08/01   87/08/01 10.01    24     24     0 YOURID
p FEB27               01.01  87/02/27   87/03/07 14.52    12     11     2 YOURID
  JAN10               01.00  87/01/10   87/01/10 17:07    21     21     0 YOURID
  JUL01               01.01  87/07/07   87/07/07 12.52    10     10     0 YOURID
  JUL14               01.00  87/07/14   87/07/14 16.15    20     20     0 YOURID
p JUN04               01.00  87/06/04   87/06/04 11.23    85     85     0 YOURID
p JUN18               01.04  87/06/18   87/06/27 08.43    34     36     6 YOURID
  **END**
```

For more information about using ISPF/PDF to print data sets, see *z/OS ISPF User's Guide Vol I*.

Printing Data Sets with the Information Center Facility

Using the UTILITY service of the Information Center Facility, you can request to print a sequential data set, or a partitioned data set with some or all of its members. Each time you print a data set or member(s) of a data set, you can select the printer you want to use.

To select a data set and a printer on which to print, display the Information Center Facility main menu panel and select UTILITY (option 10) from the list of services.

```
              TSO/E INFORMATION CENTER FACILITY USER SERVICES
Option  ===> 10

Select one of the following options.  To scroll, press UP or DOWN.

   0  DESCRIBE    - Read a short description of options on this panel
   1  NEWS        - Obtain system news
   2  NAMES       - Find a name/phone number
   3  OFFICE      - Use mail/document/other office services Facility
   4  PROGRAM     - Use program creation/execution services
   5  ANALYSIS    - Perform data analysis/report creation/decision support
   6  CHART       - Create charts/graphs
   7  COURSES     - Use education services
   8  PDF         - Use ISPF/PDF services
   9  PROBLEM     - Report problems
  10  UTILITY     - Use utility services
   I  INTRO       - Learn to use the Information Center Facility
   T  TUTORIAL    - Read descriptions of options on this panel
   X  EXIT        - Exit from the Information Center Facility


To view PF key definitions, type KEYS on COMMAND or OPTION line of any panel.
```

After selecting UTILITY, another menu panel appears. Enter P on the Option line to select the HARDCOPY option.

```
          Information Center Facility - Utilities Panel

Option ===> P

Select one of the following options.  To scroll press UP or DOWN.

    0 DESCRIBE   - Read a short description of the options on this panel
    P HARDCOPY   - Print a sequential or partitioned data set
    T TUTORIAL   - Read a detailed description of the options on this panel
    X EXIT       - Return




To view PF key definitions, type KEYS on COMMAND or OPTION line of any panel.
```

The next panel that appears is a data entry panel on which you can enter a data set name or member name and printer selection information. From this panel, you can:

- Print a sequential data set, one or more members of a partitioned data set, or an entire partitioned data set
- Display a data set selection list
- Display a printer selection list
- Specify the number of copies you want to print.

These tasks are described in the following topics.

```
                              Print Request
COMMAND ===>

DATA SET:
   Project      ===> _________   Project or owner of data set
   File Name    ===> _________   Name or part of name with * suffix
   File Type    ===> _________   Type or part of type with * suffix
   File Member  ===> _________   Member or part of member with * suffix
                                 (Optional if data set is partitioned)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   Data Set Name     ===>  ______________________________________________

PRINTER SELECTION  (if known):
   Printer Location  ===>  _________  Printer location or *
   Printer Format    ===>  _________  Document format or *
   Printer Type      ===>  _________  Printer Type or *

Number of copies     ===>  ___        Specify 1 to 255 copies
```

Printing an Entire Data Set

To print an entire data set, type the three data set qualifiers (project, file name, file type) in the DATA SET fields. Fill in the PRINTER SELECTION fields and press the Enter key.

```
                              Print Request
COMMAND ===>

DATA SET:
   Project      ===> PREFIX___   Project or owner of data set
   File Name    ===> MEMO_____   Name or part of name with * suffix
   File Type    ===> TEXT_____   Type or part of type with * suffix
   File Member  ===> _________   Member or part of member with * suffix
                                 (Optional if data set is partitioned)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   Data Set Name     ===>  ______________________________________________

PRINTER SELECTION  (if known):
   Printer Location  ===>  RNS 18S_______  Printer location or *
   Printer Format    ===>  L4SAMP________  Document format or *
   Printer Type      ===>  TEST1_________  Printer Type or *

Number of copies     ===>  1__        Specify 1 to 255 copies
```

If you need information about which printers are available for use, see “Displaying a Printer Selection List” on page 110. That topic describes how to request a list of available printers.

Displaying a Data Set Selection List

To display a list of data sets so that you may select one to print, type in a project name (prefix) and type asterisks (*) for file name and file type.

```
                              Print Request
COMMAND ===>

DATA SET:
   Project      ===> PREFIX___   Project or owner of data set
   File Name    ===> *________   Name or part of name with * suffix
   File Type    ===> *________   Type or part of type with * suffix
   File Member  ===> _________   Member or part of member with * suffix
                                 (Optional if data set is partitioned)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   Data Set Name     ===>  _____________________________________________

PRINTER SELECTION  (if known):
   Printer Location  ===>  _____________  Printer location or *
   Printer Format    ===>  _____________  Document format or *
   Printer Type      ===>  _____________  Printer Type or *

Number of copies     ===>  1__        Specify 1 to 255 copies
```

When you press the Enter key, a list of data sets with *prefix* as the first qualifier is displayed. Type S next to the data set you want to print. (You may select only one data set to print from this panel.)

```
          ICF - LIST OF FILES                                    ROW  1 OF 4
COMMAND ===>                                                  SCROLL ===> PAGE
Type S to select or D to delete next to the desired FILE.

   PROJECT   FILE NAME  FILE TYPE

_  PREFIX    MEMO       TEXT
_  PREFIX    TEST       DATA
s  PREFIX    SAMPLE     TEXT
_  PREFIX    WORK       DATA
*****************************BOTTOM OF DATA************************************
```

After you press the END key, the Print Request panel is displayed with the selected data set name filled in. Fill in the PRINTER SELECTION fields and press the Enter key to print the data set.

```
                              Print Request
COMMAND ===>

DATA SET:
   Project      ===> PREFIX___   Project or owner of data set
   File Name    ===> SAMPLE___   Name or part of name with * suffix
   File Type    ===> TEXT_____   Type or part of type with * suffix
   File Member  ===> _________   Member or part of member with * suffix
                                 (Optional if data set is partitioned)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   Data Set Name     ===>  _____________________________________________

PRINTER SELECTION  (if known):
   Printer Location  ===>  RNS 18S_______  Printer location or *
   Printer Format    ===>  L4SAMP________  Document format or *
   Printer Type      ===>  TEST1_________  Printer Type or *

Number of copies     ===>  1__        Specify 1 to 255 copies
```

If you need information about which printers are available for use, see “Displaying a Printer Selection List” on page 110. That topic describes how to request a list of available printers.

Printing One or More Data Set Members

To print one member of a data set, type in the project, file name, file type, and file member on the Print Request panel. Type in the Printer Selection information and press the Enter key.

```
                              Print Request
COMMAND ===>

DATA SET:
   Project      ===> PREFIX___   Project or owner of data set
   File Name    ===> SAMPLE___   Name or part of name with * suffix
   File Type    ===> TEXT_____   Type or part of type with * suffix
   File Member  ===> MEM1_____   Member or part of member with * suffix
                                 (Optional if data set is partitioned)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   Data Set Name     ===>  _____________________________________________

PRINTER SELECTION  (if known):
   Printer Location  ===>  RNS 18S_______  Printer location or *
   Printer Format    ===>  L4SAMP________  Document format or *
   Printer Type      ===>  TEST1_________  Printer Type or *

Number of copies     ===>  1__        Specify 1 to 255 copies
```

To print more than one data set member, display a member list from which you can select members to print. To display the member list, fill in the project, file name, and file type and type an asterisk (*) in the file member field.

```
                              Print Request
COMMAND ===>

DATA SET:
   Project      ===> PREFIX___   Project or owner of data set
   File Name    ===> SAMPLE___   Name or part of name with * suffix
   File Type    ===> TEXT_____   Type or part of type with * suffix
   File Member  ===> *________   Member or part of member with * suffix
                                 (Optional if data set is partitioned)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   Data Set Name     ===>  _____________________________________________

PRINTER SELECTION  (if known):
   Printer Location  ===>  RNS 18S_______  Printer location or *
   Printer Format    ===>  L4SAMP________  Document format or *
   Printer Type      ===>  TEST1_________  Printer Type or *

Number of copies     ===>  1__        Specify 1 to 255 copies
```

When you press the Enter key, you see a list of members and information about each member. To select members for printing, type an S next to the members you want printed. Press the END key once and the Member List panel will be re-displayed. This allows you to confirm your request to print the members. Press the END key again to print all of the selected members. If you have not yet filled in a printer name, when you press the END key you will be prompted for a printer name.

If you need information about which printers are available for use, see “Displaying a Printer Selection List” on page 110. That topic describes how to request a list of available printers.

```
------------------------------- Member List ---------------- ROW  1 OF 5  00020
COMMAND ===>                                                  SCROLL ===> PAGE

Type S in the S column to select a member or members for printing.
To exit and print selections, press END.  To  exit without printing type CANCEL.

S  Member          VV.MM   Created    Last Modified  Size   Init   Mod  ID
   MEM1            01.02  89/03/30   89/04/05 15:59    10     21     2   PREFIX
   MEM3            01.01  89/03/30   89/04/03 14:07     1      1     0   PREFIX
s  MEM4            01.03  89/04/03   89/04/03 14:21     1      1     0   PREFIX
s  NEWMEM          01.02  89/04/03   89/04/04 07:21     1      1     0   PREFIX
s  TESTMEM         01.02  89/04/05   89/04/05 14:14     1      1     0   PREFIX
**********************************BOTTOM OF DATA********************************
```

Displaying a Printer Selection List

When you want to print information, but are unsure which printers are available for your use, you can display a list of printers. Type an asterisk (*) in each of the three PRINTER SELECTION fields and press the Enter key. (This can be done either before or after the DATA SET fields are filled in.)

```
                              Print Request
COMMAND ===>

DATA SET:
   Project      ===> PREFIX___   Project or owner of data set
   File Name    ===> TEST_____   Name or part of name with * suffix
   File Type    ===> DATA_____   Type or part of type with * suffix
   File Member  ===> _________   (Optional if data set is partitioned)

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   Data Set Name     ===>  ____________________________________________

PRINTER SELECTION  (if known):
   Printer Location  ===>  *_____________  Printer location or *
   Printer Format    ===>  *_____________  Document format or *
   Printer Type      ===>  *_____________  Printer Type or *

Number of copies     ===>  1__        Specify 1 to 255 copies
```

A panel is displayed that lists the printers you can select for printing. Follow the instructions on the panel to choose a printer. Press END to return to the Print Request panel with the PRINTER SELECTION information filled in.

Chapter 12. Deleting Data Sets

Deleting a data set means deleting access to the data set. When a data set is cataloged, deleting it means removing the data set's catalog entry. When a data set is on a particular volume, deleting it means removing the volume identifier.

You can delete a data set in the following ways:

- With the DELETE command
- From the UTILITIES option of ISPF/PDF
- With the DELETE operand used with the ALLOCATE and FREE commands. For information about how to use this operand, see "KEEP and DELETE Operands" on page 42 and "Releasing Data Sets and Specifying Their Disposition" on page 61.

Deleting Data Sets with the DELETE Command

To delete one or more data sets or one or more members of a partitioned data set from the system, use the DELETE command.

When the system deletes an entire data set, it removes the catalog entry. The system deletes members of a partitioned data set by removing the member name from the directory of the partitioned data set.

When you delete a member that has one or more aliases, the system does not delete the aliases for that member. When you delete an alias, the system does not delete its associated member. Thus you must explicitly delete both a member and its alias or aliases.

Using the DELETE command, you can specify:

- The name of the data set you want to delete. You can also specify a data set's password, if it is password-protected.
- The name of the DD statement that identifies either the volume on which the data set to be deleted resides or the data set itself (FILE operand)
- The name of the catalog that contains the data sets you want to delete (CATALOG operand)
- Whether to delete a data set depending on its retention period (PURGE/NOPURGE operands)
- Whether to scratch (remove) a non-VSAM data set's entry from the volume table of contents (VTOC) on the volume on which the data set resides (SCRATCH/NOSCRATCH operands)
- That the system is to delete an alias entry (ALIAS operand)

Deleting a Data Set

You can delete a data set by specifying either:

- The data set name
- The data set name and its password
- Multiple data set names
- The volume identifier from the ddname (file) associated with the data set

Deleting a Single Data Set

To delete a data set that doesn't have a password, enter the DELETE command and specify the data set name.

© Copyright IBM Corp. 1988, 2024

Example

To delete data set PREFIX.PARTS.DATA, enter:

```
DELETE parts.data
```

Deleting a Password-Protected Data Set

To delete a data set that is password-protected, enter the DELETE command:

- Specify the data set name
- Immediately enter a slash, followed by the data set's password

Example

To delete data set PREFIX.PARTS.DATA with the password BLOBULNT, enter:

```
DELETE parts.data/blobulnt
```

For information about password-protecting data sets, see the PROTECT command in *z/OS TSO/E Command Reference*.

Deleting More than One Data Set

To delete a number of data sets, enter the DELETE command:

- Specify each data set name separated by a comma or blank
- Enclose the group of names in parentheses

Example

To delete data sets PREFIX.PARTS.DATA, PREFIX.PARTS.TEXT, PREFIX.PARTS.ASM, and PREFIX.PARTS.CNTL, enter:

```
DELETE (parts.data,parts.text,parts.asm,parts.cntl)
```

FILE Operand

To delete PREFIX.PARTS.DATA using the volume identifier from the ddname (file) associated with the data set, enter the DELETE command and specify:

- The data set name
- The FILE operand

Example

To delete data set PREFIX.PARTS.DATA if the file SYSIN contains the volume identifier on which the data set PREFIX.PARTS.DATA resides, enter

```
DELETE parts.data FILE(sysin)
```

Deleting a Data Set Entry from a Catalog

CATALOG Operand

To delete a data set entry from a catalog, enter the DELETE command:

- Specify the CATALOG operand
- Enclose the name of the catalog in parentheses

Example

To delete data set PREFIX.PARTS.DATA from the catalog USERCAT1, enter:

```
DELETE parts.data CATALOG(usercat1)
```

When deleting an entry for a password-protected data set, you may either specify the password for the data set itself, or specify the password to the catalog that contains the entries you want to delete.

Example

If catalog USERCAT1 had a master password of PFEFFER, you could specify:

```
DELETE parts.data CATALOG(usercat1/pfeffer)
```

Deleting a Data Set Based On Its Retention Period

PURGE Operand

When you allocated a particular data set, you may have specified a retention period after which the system would automatically delete the data set. To delete a data set regardless of its retention period, use the PURGE (abbreviation PRG) operand.

Example

If data set PREFIX.PARTS.DATA still has 30 days left in its retention period, but you want to delete it now, enter:

```
DELETE parts.data PURGE
```

NOPURGE Operand

The NOPURGE operand (abbreviation NPRG) specifies that the system should delete a data set only if its retention period has expired. NOPURGE is a default with the DELETE command.

Deleting and Scratching a Data Set's VTOC Entry

The volume table of contents (VTOC) is a data set that describes the contents of a direct access volume. It is used to account for each data set and available space on the volume. The VTOC includes the name, location, organization, and other control information about each data set stored on the volume.

SCRATCH Operand

When you delete a data set, you remove its entry from the volume table of contents (VTOC) on the volume on which it resides. The SCRATCH operand is a default on the DELETE command.

NOSCRATCH Operand

If you do not want to remove an entry from the VTOC, enter the DELETE command and specify:

- The data set name
- The NOSCRATCH (abbreviation NSCR) operand.

Example

To delete data set PREFIX.PARTS.DATA, but not remove its entry from the VTOC, enter:

```
DELETE parts.data NOSCRATCH
```

If you use the NOSCRATCH operand with the DELETE command, you remove the catalog entry for the data set. The data set still exists, and to reaccess it, you must specify the volume on which it resides.

Deleting an Alias Entry

ALIAS Operand

To delete an alias entry, issue the DELETE command and specify:

- The alias name of the member of the partitioned data set
- The ALIAS operand.

Example

To delete alias entry, PREFIX.JCL.CNTL(ALIAS1), enter:

```
DELETE jcl.cntl(alias1) ALIAS
```

For more information about the DELETE command, see *z/OS TSO/E Command Reference*.

Using ISPF/PDF to Delete a Data Set

To delete a sequential data set, a partitioned data set, or members of a partitioned data set, you can use the UTILITIES option of ISPF/PDF.

To delete a data set, select the ISPF/PDF UTILITIES option (option 3) from the ISPF/PDF Primary Option Menu.

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 3
                                                            USERID    - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters    TIME      - 12:47
    1  BROWSE      - Display source data or output listings  TERMINAL - 3277
    2  EDIT        - Create or change source data            PF KEYS   - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

Deleting an Entire Data Set

To delete an entire data set, select the DATASET option (option 2).

```
-------------------------  UTILITY SELECTION MENU  ----------------------------
OPTION ===> 2

    1  LIBRARY     - Compress or print data set.  Print index listing.
                       Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                       display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                       Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

The next panel you see allows you to do several data set management tasks. To delete a data set, type D for delete on the OPTION line. Then specify the data set name in the ISPF LIBRARY fields or in the OTHER PARTITIONED OR SEQUENTIAL DATA SET field. Refer to “Specifying a Data Set Name” on page 54 for information about specifying the data set name in these fields.

```
------------------------------  DATA SET UTILITY  -----------------------------
OPTION ===> d

    A - Allocate new data set                 C - Catalog data set
    R - Rename entire data set                U - Uncatalog data set
    D - Delete entire data set                S - Data set information (short)
    blank - Data set information

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  oldtest
    TYPE    ===>  data

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>          (If not cataloged, required for option "C")

DATA SET PASSWORD ===>          (If password protected)
```

When you press the Enter key, you see the Confirm Delete panel. The Confirm Delete panel requires you to repeat the delete request to insure that you didn't type the D by mistake. To delete the data set, press the Enter key. To cancel the delete process, press the END PF key.

```
------------------------------  CONFIRM DELETE  -------------------------------
COMMAND ===>

DATA SET NAME:  PREFIX.OLDTEST.DATA
VOLUME:         TSO018
CREATION DATE:  87/02/15


INSTRUCTIONS:

    Press ENTER key to confirm delete request.
       (The data set will be deleted and uncataloged.)

    Enter END command to cancel delete request.
```

Deleting One or More Members of a Data Set

To delete one or more members of a partitioned data set, select the UTILITIES option (option 3) from the ISPF/PDF Primary Option Menu. Then from the Utilities Selection menu, select the LIBRARY option (option 1).

```
-------------------------  UTILITY SELECTION MENU  ----------------------------
OPTION ===> 1

    1  LIBRARY     - Compress or print data set.  Print index listing.
                     Print, rename, delete, or browse members
    2  DATASET     - Allocate, rename, delete, catalog, uncatalog, or
                     display information of an entire data set
    3  MOVE/COPY   - Move, copy, or promote members or data sets
    4  DSLIST      - Print or display (to process) list of data set names
                     Print or display VTOC information
    5  RESET       - Reset statistics for members of ISPF library
    6  HARDCOPY    - Initiate hardcopy output
    8  OUTLIST     - Display, delete or print held job output
    9  COMMANDS    - Create/change an application command table
   10  CONVERT     - Convert old format messages/menu panels to new format
   11  FORMAT      - Format definition for formatted data Edit/Browse
   12  SUPERC      - Compare data sets (Standard dialog)
   13  SUPERCE     - Compare data sets (Extended dialog)
   14  SEARCH-FOR  - Search data sets for strings of data
```

The next panel you see allows you to do many different tasks with data set members. To delete one member of a data set, type D on the OPTION line of the Library Utility panel and specify the three data set qualifiers plus the member name in the ISPF LIBRARY fields. When you press the Enter key, the member is deleted.

```
------------------------------  LIBRARY UTILITY  -----------------------------
OPTION ===> d

    blank - Display member list              B - Browse member
    C - Compress data set                    P - Print member
    X - Print index listing                  R - Rename member
    L - Print entire data set                D - Delete member
    I - Data set information                 S - Data set information (short)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  report      ===>             ===>            ===>
    TYPE    ===>  text
    MEMBER  ===>  yearend       (If option "P", "R", "D", or "B" selected)
    NEWNAME ===>                    (If option "R" selected)


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>              (If not cataloged)

DATA SET PASSWORD ===>              (If password protected)
```

To delete more than one member of a data set, you must display a member list from which you can select members. To display the member list, leave the OPTION line blank on the Library Utility panel. Then specify the three data set qualifiers in the ISPF LIBRARY field and leave the MEMBER field blank.

```
------------------------------  LIBRARY UTILITY  ----------------------------
OPTION ===>

    blank - Display member list              B - Browse member
    C - Compress data set                    P - Print member
    X - Print index listing                  R - Rename member
    L - Print entire data set                D - Delete member
    I - Data set information                 S - Data set information (short)

ISPF LIBRARY:
    PROJECT ===>  prefix
    GROUP   ===>  report      ===>             ===>            ===>
    TYPE    ===>  text
    MEMBER  ===>                    (If option "P", "R", "D", or "B" selected)
    NEWNAME ===>                    (If option "R" selected)


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
    VOLUME SERIAL ===>              (If not cataloged)

DATA SET PASSWORD ===>              (If password protected)
```

You then see a list of member names and information about each member. To delete members, type d next to the members you want deleted. (Notice that you can also browse, rename, and print members from this option.) When you press the Enter key, the word '*DELETED' appears next to the members you selected and the member is deleted. When you are finished deleting members, press the END PF key.

```
LIBRARY - PREFIX.REPORT.TEXT ------------------------------- ROW 00001 OF 00008
COMMAND ===>                                                  SCROLL ===> PAGE
   NAME     RENAME    VV.MM   CREATED       CHANGED      SIZE   INIT   MOD  ID
  PROJ1               01.02  87/04/13   87/05/21 09.55    44     39     4 YOURID
d PROJ2               01.00  87/08/01   87/08/01 10.01    24     24     0 YOURID
d PROJ3               01.01  87/02/27   87/03/07 14.52    12     11     2 YOURID
  PROJ4               01.00  87/01/10   87/01/10 17:07    21     21     0 YOURID
  PROJ5               01.01  87/07/07   87/07/07 12.52    10     10     0 YOURID
  PROJ6               01.00  87/07/14   87/07/14 16.15    20     20     0 YOURID
d PROJ7               01.00  87/06/04   87/06/04 11.23    85     85     0 YOURID
d YEAREND             01.04  87/06/18   87/06/27 08.43    34     36     6 YOURID
  **END**
```

For more information about deleting data sets in ISPF/PDF, see *z/OS ISPF User's Guide Vol I*.

Part 3. Running a Program

Before a program can run (execute), it must be prepared for execution. If you are not a programmer, you might be given a program that has been previously prepared. If you are a programmer, see *z/OS TSO/E Programming Guide* for more information about the following steps used to prepare a program for execution.

1. Compiling Source Code

   The program's source code must be changed into machine language, a format that the processor can interpret. This change to machine language is called compiling and it is done by a compiler. The compiler creates data sets called *object modules* that contain the compiled code.

2. Link-Editing Object Modules

   Some programs require data from system libraries or from other programs to complete execution. When this data is in object module form, you can link-edit a number of object modules together using TSO/E. A group of object modules linked together is known as a **load module**.

3. Loading Programs

   Most programs reside on auxiliary storage, but must be in main storage to run. Placing a program into main storage is called *loading*.

You can run a program two ways in TSO/E:

1. Foreground Processing - You can issue commands at your terminal to run programs and control the program's process from your terminal. As the program is running, you do not have the use of your terminal for other work.
2. Background or Batch Processing - If you have the proper authority, you can use JCL statements to set up the proper conditions for running your program, and then use the SUBMIT command to submit the JCL statements to the processor. The program is put on a queue and executed at a later time, leaving your terminal free to do other work.

This part describes how to:

- Run programs in the foreground - (Chapter 13, “Running Programs in the Foreground,” on page 121)
- Submit and monitor a background job - (Chapter 14, “Submitting and Monitoring a Background Job,” on page 123)
- Direct the output of a background job - (Chapter 15, “Processing the Output of a Batch Job,” on page 133)
- Issue TSO/E commands in a background environment - (Chapter 16, “Executing Foreground Commands from a Background Job,” on page 141)

© Copyright IBM Corp. 1988, 2024

Chapter 13. Running Programs in the Foreground

Two TSO/E commands run a program in the foreground:

- RUN command - If your installation has certain program products, you need only issue the RUN command and the program products automatically compile, link-edit, load, and execute the program. For information about the RUN command, see *z/OS TSO/E Programming Guide* and *z/OS TSO/E Command Reference*.
- CALL command - If a program has been previously prepared and is ready to run, you need only issue the CALL command to load and execute it.

Executing a Program with the CALL Command

To load and execute a program that exists in executable (load module) form, use the CALL command. The program may be user-written or it may be owned by the system, for example, a compiler, sort, or utility program.

You must specify the name of the program (load module), which must be a member of a partitioned data set. Also, you can pass parameters to the program.

Loading and Executing Load Modules

To load and execute a load module, specify the name of the data set with the name of the member enclosed in parentheses.

Example

To load and execute the load module in member SCAN of data set PUBS.LOAD, enter:

```
CALL pubs(scan)
```

If SCAN is the only member of data set PUBS.LOAD, you need not specify the member name when issuing the CALL command.

```
CALL pubs
```

If the partitioned data set does not conform to data set naming conventions described in “TSO/E Data Set Naming Rules and Conventions” on page 20, then you must specify the member name that contains the program you want to execute. If you specify a fully-qualified data set name, enclose it in single quotation marks in the following manner:

```
CALL 'user5.myprog.loadmod(discharg)'
```

or

```
CALL 'sys1.linklib(ieuasm)'
```

Passing Parameters when Loading and Executing Load Modules

To load and execute the load module in a partitioned data set and pass it parameters, specify the parameters enclosed in single quotation marks following the data set name.

© Copyright IBM Corp. 1988, 2024

Example

To pass **first** as a parameter to the load module in member MYPROG of data set PROGRAM.LOAD, enter:

```
CALL program(myprog) 'first'
```

To pass **first, second,** and **third** as parameters to member MYPROG, enter:

```
CALL program(myprog) 'first,second,third'
```

For more information about the CALL command, see *z/OS TSO/E Command Reference*.

Chapter 14. Submitting and Monitoring a Background Job

Some TSO/E tasks require a great deal of system resources, or keep the user from doing other TSO/E tasks until one particular task is completed. To increase your productivity and improve system performance, you can submit such tasks to run in the background. Jobs executed in the background are also known as **batch jobs**, which means they execute in a batch environment. Hereafter in this chapter, we'll use the terms batch job and batch environment to refer to jobs running in the background.

Batch jobs consist of:

- Job control language (JCL) statements
- The user's program
- The data for the program

You must first prepare the JCL and then submit the job. The system executes the job and directs the results to an output queue.

The JCL for a batch job can be user-created under EDIT, or system-created using the SUBMIT command. User-created JCL is discussed in Chapter 16, “Executing Foreground Commands from a Background Job,” on page 141. Generally, the JCL data set should have the descriptive qualifier CNTL.

When the job is submitted, the system gives it a job number, which you can use to identify the job. Using the job number, you can display the status of the job to see what stage of processing it is in. You can request the system to notify you when the job terminates. If you want to stop the processing of your batch job, you can cancel the job.

Use the SUBMIT, STATUS, CANCEL and OUTPUT commands primarily to control the submission and processing of jobs in a batch environment.

Note: If your installation uses security labels and security checking, you can submit a batch job to run at a security label *that is greater than the security label* you are logged on at, provided you are authorized to use the job's security label. However, during your current TSO/E session, you will not be able to use the OUTPUT command to process the output produced by the job or the CANCEL command to cancel the job. If the output is held, you can process it by logging on with a security label that is greater than or equal to the job's security label.

Depending on the security options used at your installation, you may not be able to submit a job at a security label that is *less* than the security label you are currently logged on with.

Submitting Batch Jobs

If your installation authorizes you to submit batch jobs for processing, the system lets you use the four commands SUBMIT, STATUS, CANCEL, and OUTPUT that control the processing of batch jobs. You can use these commands to submit a batch job, display the status of a batch job, cancel a batch job, and control the output of a batch job.

The JOB Statement

Before you submit a batch job with the SUBMIT command, you must know which data set (or member of a partitioned data set) contains the job or jobs you want to submit. Each job consists of job control language (JCL) statements, program instructions, and data.

The first JCL statement in the data set is usually a JOB statement. The job name in the JOB statement can be up to eight characters long and should consist of your user ID followed by one or more letters or numbers, for example, **YOURIDA** or **YOURID5**.

© Copyright IBM Corp. 1988, 2024

If the job name does not begin with your user ID, you can submit the job with the SUBMIT command and request its status with the STATUS command. However, you cannot refer to the job with the CANCEL or OUTPUT command unless your system uses an installation-written exit routine permitting this. If your system uses the default IBM-supplied exit routine, the CANCEL and OUTPUT commands will be unable to refer to the job.

If the job name consists of only your user ID, the system prompts you for one or more characters to complete the job name. This allows you to change job names without re-editing the data. For example, you may submit the same job several times, and supply a different character for the job name each time you are prompted.

If the first JCL statement of your data set is not a JOB statement, the system generates the following JOB statement when you submit the job with the SUBMIT command:

```
//userid JOB accounting info,
//   userid, ** JOB STATEMENT GENERATED BY SUBMIT **
//   NOTIFY=userid,
//   MSGLEVEL=(1,1)
```

TSO/E prompts you for a character to complete the job name. The job accounting information is the information specified by the user when logging on to the system. With no JOB statement in the data set, the security label assigned to the job (if your installation uses security labels) is the security label you are logged on at.

If there is a JOB statement for the job you are submitting, and the JOB statement contains a SECLABEL operand, then the job will run using that security label. If the JOB statement for the batch job does not contain a SECLABEL operand, the job will run at the security label you are logged on at.

If SECLABEL is present on the JOB statement and the security label specified is greater than the one you are logged on with, the job will run in the background, but you will *not* be able to cancel the job (CANCEL command) or process the output from the job (OUTPUT command) during your current session. You have to log off and log back on to TSO/E at a security label equal to or greater than the job's security label.

As a practice, you should examine a batch job's JOB statement before submitting the job to verify the SECLABEL (if any) associated with the job.

When you enter the SUBMIT command, you must give the name of a data set (or data sets) containing the batch job (or jobs). You can also use the NONOTIFY operand to specify that you do not want to be notified when the batch job with a generated JOB statement terminates. The SUBMIT command performs best if you enter the fully-qualified data set name in quotation marks. Submitted data sets must have a logical record length of 80 bytes, a record format of fixed-blocked (FB), and must not contain lowercase characters.

Submitting a Batch Job with the SUBMIT Command

To submit one or more batch jobs for processing, use the SUBMIT command. Each job submitted must reside in either a sequential data set, a direct-access data set, or in a member of a partitioned data set, unless you use the asterisk (*) function of the SUBMIT command. The asterisk (*) function tells the system that you will enter the data in some way other than a permanent data set. For more information on the asterisk function, see “The SUBMIT * Function” on page 128. Submitted data sets must contain fixed-blocked, 80-byte records.

Any of these data sets can contain part of a job, one job, or more than one job that can be executed via a single entry of SUBMIT. Each job must comprise an input job stream, that is, JCL plus data. Do not submit data sets with descriptive qualifiers TEXT or PLI if the characters in these data sets are lowercase.

You may include more than one job in one data set. You can omit the JOB statement for the first job, but all jobs after the first must have their own JOB statement. Although you submit all jobs in the data set with one SUBMIT command, you can subsequently refer to each job with separate STATUS, CANCEL, and OUTPUT commands.

When you submit more than one job with a single command and TSO/E finds an error while processing the first job, the second job is not processed. An error that occurs in the second job does not affect the first.

Any jobs processed before the error are submitted for execution; jobs that were not processed because of the error should be resubmitted after the error is corrected. Similarly, a syntax error that occurs after the end of a single job is ignored.

If you wish to provide a JOB statement but you also want to be prompted for a unique job name character, put your user ID in the job name field and follow it with blanks so that there is room for the system to insert the prompted-for character. This process allows you to change job names without re-editing the JCL data set.

After SUBMIT has successfully submitted a job for conventional batch processing, it issues a 'job name(job ID) submitted' message. The job ID is a unique job identifier assigned by the job entry subsystem (JES).

If SUBMIT is to generate a JOB statement preceding one or more JES control statements, make the first statement of your data set a comment. If this is not done, SUBMIT generates the JOB statement following any JES control statements.

Note: If the LOGON command is found in the SUBMIT job stream before a JOB statement is found, SUBMIT processing uses the LOGON command to build the JOB and EXEC statements. The job created (if the PROC operand was not on the LOGON command) is the TSO/E terminal monitor program (IKJEFT01) running in the background. All input after the LOGON command is considered to be TSO/E commands.

If the SECLABEL operand appears on the LOGON command, SUBMIT processing includes the SECLABEL keyword with its value on the JOB statement it builds. The batch job will then run with the security label on the JOB statement.

More information on submitting TSO/E commands in the background is in “Submitting Commands Using the SUBMIT Command” on page 141.

Data sets that are dynamically allocated by the SUBMIT command are not automatically freed when the command ends. You must explicitly free dynamically allocated data sets using the FREE command.

A submitted data set need not contain an entire job. A JCL data set and a source data set could be used if both were the proper type of data set.

Using the SUBMIT command, you can specify:

- The data set or data sets that contain the input stream you want to process as a batch job
- The data set or data sets that contain the input data you want to use when processing a batch job
- Whether you want the system to hold the job's output for later output processing (HOLD/NOHOLD operands)
- Whether you want to predefine characters that the system appends to the job's name (JOBCHAR/NOJOBCHAR operands)
- Whether you want the system to prompt you for a password when it generates a job statement (PASSWORD/NOPASSWORD operands)
- Whether you want to define a user ID that the system uses when it generates a JOB statement (USER/NOUSER operands)
- Whether you want the system to let you know when your job has terminated (NOTIFY/NONOTIFY operands)
- That you want the option of continuing or cancelling the job after the job stream has been read into the system (* PAUSE operand)
- A character string that indicates the end of the job stream (* END operand)

Submitting One Job

To submit as a batch job one member of a data set, specify the name of the data set followed by the name of the member enclosed in parentheses.

Example

To submit the job in the member JOB1 in data set JCL.CNTL, enter:

```
SUBMIT jcl(job1)
```

To submit a number of jobs at once, specify the list of data set names in parentheses, with each data set name separated from the others by a comma.

Example

To submit the jobs in data sets JOB5.CNTL, USERID.SIX.JOB, and member JCL.CNTL(JOB1), enter:

```
SUBMIT (job5,'userid.six.job',jcl(job1))
```

You can specify the data sets a job uses for input when you issue the SUBMIT command. To identify the input data set, specify the input data set's name following the data set name that contains the job. For this form of the command to be valid, the input data set must have a descriptive qualifier of DATA.

Example

To run the job in data set JOB5.CNTL, and to use the input data in data set MYDATA.DATA, enter:

```
SUBMIT (job5,mydata)
```

Submitting More than One Job

To submit a number of jobs at once, and to specify the data sets from which the jobs will get their input, issue the SUBMIT command and enclose in parentheses:

- The name of the data set that holds the JCL
- The name of the data set that contains the input (or data)

Example

To submit the jobs in data sets JOB5.CNTL and JOB7.CNTL, where data sets FIVEIN.DATA and SEVENIN.DATA contain input for each job, enter:

```
SUBMIT (job5,fivein,job7,sevenin)
         |       |     |       |
        job   input   |       |
                    job   input
```

Holding a Batch Job's Output

HOLD/NOHOLD Operands

To have the system hold a job's output for later processing, use the HOLD operand. NOHOLD specifies that you do not want the output held for later processing. NOHOLD is the default.

Example

To have the system retain the output from the job in data set JOB5.CNTL, enter:

```
SUBMIT job5 HOLD
```

Appending Characters to a Batch Job's Job Name

JOBCHAR/NOJOBCHAR Operands

To have the system append characters to the job name on each job statement in a data set, use the JOBCHAR operand. This operand is useful if you want to submit the same job several times but each time use a different job name. Use only one character if you plan to use the STATUS command and the job name matches your user ID. Issue the SUBMIT command and specify:

- The name of the data set containing the job(s)
- The JOBCHAR operand with the characters to be appended to each job's name enclosed in parentheses

Example

To define the characters *XM* as the characters that the system appends to each job's name in data set JOB5.CNTL, enter:

```
SUBMIT job5 JOBCHAR(xm)
```

To have the system prompt you for job name characters whenever the job name is the user ID, use the NOJOBCHAR operand. NOJOBCHAR is a default with the SUBMIT command.

Password Prompting When Submitting a Batch Job

PASSWORD/NOPASSWORD Operand

To request that the system prompt you for a password when generating a job statement, use the PASSWORD operand. This operand is valid only if you have RACF installed, and you request the system to generate a JOB statement for you.

Example

To request that the system prompt you for a password when it generates a job statement for the job in data set JOB5.CNTL, enter:

```
SUBMIT job5 PASSWORD
```

If you do not have RACF installed at your installation, the NOPASSWORD operand, specifying that the system not prompt you for a password, is a default.

Specifying a User ID When Submitting a Batch Job

USER/NOUSER Operand

To use a specific user ID when the system generates a JOB statement, use the USER operand with the user ID enclosed in parentheses.

Example

To define user ID YOURID as the user ID the system uses when it generates a JOB statement for the job in data set JOB5.CNTL, enter:

```
SUBMIT job5 USER(yourid)
```

This operand is valid only if you have RACF installed, and you request the system to generate a JOB statement for you. The user ID you specify with this operand is also used as the job name for the job you submit. If you have RACF installed, this operand is the default.

To keep the system from defining either a PASSWORD or a user ID when it generates a JOB statement, use the NOUSER operand.

Example

If you do not want the system to build either a PASSWORD or user ID when you run the job in data set JOB5.CNTL, enter:

```
SUBMIT job5 NOUSER
```

NOUSER is the default if you do not specify either USER or NOUSER and if you do not have RACF installed.

Receiving Notice When a Batch Job is Done

NOTIFY/NONOTIFY Operand

The system usually notifies you when a batch job has ended, because the NOTIFY operand is the default with the SUBMIT command. The system ignores the NOTIFY operand if the data set you submit already contains a JOB statement.

To prevent the system from notifying you when a batch job has ended, use the NONOTIFY operand.

Example

If you do not want the system to notify you when the job in member JOB1 in data set JCL.CNTL has ended, enter:

```
SUBMIT jcl.cntl(job1) NONOTIFY
```

The SUBMIT * Function

The SUBMIT command supports an asterisk (*) for the positional operand value and two keyword operands, END and PAUSE. The keyword operands, END and PAUSE, are valid only when (*) is specified and when the SUBMIT (*) command is not issued in EDIT mode.

SUBMIT * allows the job stream source to reside in other than a permanent data set, such as a terminal, in-storage lists, temporary data sets, and the CLIST-type in-storage lists. The job stream may be entered directly without creating and editing a data set. Figure 4 on page 129 illustrates how the SUBMIT * function is used to submit batch jobs. Note that the existing SUBMIT * function of EDIT continues to select the current data set as the input job stream. Therefore, this SUBMIT * function is not available in EDIT mode.

```
/*Example 1: Submitting a job with input from a terminal*/
READY
SUBMIT *
ENTER INPUT JOB STREAM:
//step       exec  pgm=somepgm
( null line )
ENTER JOBNAME CHARACTER(s)
a
JOB USERIDA (JOB00007) SUBMITTED
READY
/*Example 2: Submitting jobs with input from a CLIST*/
/* The following is a listing of the CLIST about to be
   submitted*/

PROC 0 STEP(2)
CONTROL PROMPT
SUBMIT * PAUSE END(GO)
//USERIDA    JOB   MSGLEVEL=1
//STEPA1     EXEC  PGM=YOURPGM
//SYSPRINT   DD    SYSOUT=A
IF &STEP=2 THEN DO
//STEPA2     EXEC  PGM=PROGRAM2
//SYSPRINT   DD    SYSOUT=A
END
ELSE DO
//STEPA3     EXEC  PGM=PROGRAM3
//SYSPRINT   DD    SYSOUT=A
END
//USERIDB    JOB   MSGLEVEL=1
//STEPB      EXEC  PGM=SOMEPGM
//SYSPRINT   DD    SYSOUT=A
GO

/* The following shows the CLIST being executed.

exec myclist list
ENTER INPUT JOB STREAM:
SUBMIT * PAUSE END(GO)
//USERIDA    JOB   MSGLEVEL=1
//STEPA1     EXEC  PGM=YOURPGM
//SYSPRINT   DD    SYSOUT=A
//STEPA2     EXEC  PGM=PROGRAM2
//SYSPRINT   DD    SYSOUT=A
//USERIDB    JOB   MSGLEVEL=1
//STEPB      EXEC  PGM=SOMEPGM
//SYSPRINT   DD    SYSOUT=A
go
SHOULD INPUT JOB STREAM BE SUBMITTED? ENTER YES OR NO: +
yes
JOB USERIDA(JOB00008) SUBMITTED
JOB USERIDB(JOB00009) SUBMITTED
READY
```

*Figure 4. The SUBMIT * Function*

Ending a Batch Job

PAUSE Operand

To request the option of continuing or cancelling a job after it has been read into the system, use the PAUSE operand.

Example
To request the option of continuing or cancelling a submitted job, enter: `SUBMIT * PAUSE`

The PAUSE operand supports the SUBMIT * function *only*. If you do not specify this operand, the system processes the job stream when it detects the end of the job stream.

END Operand

To specify a character that indicates the end of a job stream, use the END operand with the character string enclosed in quotation marks.

Example

If you want the job to end when the system encounters the character string **#$**, enter:

```
SUBMIT * END(#$)
```

The END operand supports the SUBMIT * function *only*. You can only specify alphabetic, numeric, or the special characters $, #, and @ when using this operand. If you don't specify this operand, a null or blank line indicates the end of the job stream. Specifying this operand allows you to enter blank lines as part of your job stream. The END characters must begin in column 1 and be the only data on the line.

For more information on the SUBMIT command, see *z/OS TSO/E Command Reference*.

Submitting a Batch Job from ISPF/PDF

You can submit data you are editing with the full-screen ISPF/PDF editor by issuing the SUBMIT command on the COMMAND line of the edit panel. The entire member or sequential data set is then submitted as a batch job.

Example

To submit the job you are editing with the ISPF editor, first save any changes you made, then enter the SUBMIT command on the COMMAND line of the edit panel:

```
EDIT ---- PREFIX.JCL.CNTL(JOB1) ----------------------- COLUMNS 001 072
COMMAND ===> submit                        SCROLL ===> PAGE
****** ************************ TOP OF DATA ***************************
000001 //USERIDA      JOB    MSGLEVEL=1
000002 //STEPA1       EXEC   PGM=YOURPGM
⋮
```

For more information about submitting a batch job from ISPF/PDF, see *z/OS ISPF Edit and Edit Macros*.

Allowing Another User to Submit Your Job

If you meet certain RACF requirements you can define other TSO/E users to run your jobs from their user ID without them knowing your logon password. This process is called *surrogate job submission*, and the other users are called *surrogate users*.

Be aware that if you define a surrogate user to run your jobs under your user ID, the surrogate user also has access to many other data sets and pieces of information related to your user ID. Check with your security administrator before defining surrogate users.

Before you select another user to be your surrogate, ask your RACF security administrator to make all of the necessary RACF authorizations for both your user ID and the surrogate's user ID.

See *z/OS Security Server RACF General User's Guide* for information about the RACF commands needed to define a surrogate user.

Displaying the Status of a Batch Job with the STATUS Command

After you submit a batch job, you can use the STATUS command to display its status at your terminal. The display tells you whether the job is awaiting execution, is currently executing, or has executed but is still on the output queue. The display also indicates whether a job is in hold status.

You can obtain the status of all your batch jobs, of specific batch jobs, or of a single batch job. Using the STATUS command, you can specify which of your job's or jobs' status you want to display.

You can use this command only if the installation management has given you the authority to do so.

Displaying the Status of All Your Jobs

To display the status of all the jobs whose job names consist of your user ID plus one character, issue the STATUS command with no operands.

Example

If you submitted ten jobs and you wanted to know the status of each, enter:

```
STATUS
```

Displaying the Status of Specific Jobs

To display the status of all the jobs associated with a specific job name, specify the job name following the STATUS command.

Example

To display the status of all the jobs associated with job name YOURID2, enter:

```
STATUS yourid2
```

To display the status of all the jobs associated with a number of job names, specify the list of job names, separated by commas, and enclosed in parentheses.

Example

To display the status of all the jobs whose job names are YOURID2, YOURID5, and YOURIDX, enter:

```
STATUS (yourid2,yourid5,youridx)
```

To display the status of a specific job associated with a job name, enter the job name with the system-supplied job identifier enclosed in parentheses.

Example

To display the status of the job identified by number JOB01098 with job name YOURIDX, enter:

```
STATUS youridx(job01098)
```

For more information about the STATUS command, see *z/OS TSO/E Command Reference*.

Cancelling a Batch Job with the CANCEL Command

To halt processing of batch jobs that you submitted from your terminal, use the CANCEL command. If you cancel the job successfully, the system displays a READY message at your terminal. The system also notifies the system operator when you cancel a job.

Using the CANCEL command, you can specify:

- Which job or jobs you want to cancel
- Whether you want to purge a job's output

You can use this command only if the installation management has given you the authority to do so. Also, if your installation uses security labels and security checking, you can only cancel jobs if the security label you are logged on at is equal to or greater than the security label associated with the job.

Cancelling Specific Jobs

To cancel the job associated with a specific job name, specify the job name following the CANCEL command.

Example

To cancel the job associated with job name YOURIDX, enter:

```
CANCEL youridx
```

If there is more than one job associated with job name YOURIDX, the system prompts you for a specific job identifier.

To cancel each job associated with a number of job names, specify the list of job names enclosed in parentheses and each separated from the others by a comma or blank.

Example

To cancel the jobs whose names are YOURID3, YOURID5, and YOURIDX, enter:

```
CANCEL (yourid3,yourid5,youridx)
```

To cancel a specific job associated with a job name, specify the job name with the system-supplied job identifier enclosed in parentheses.

Example

To cancel the job identified by number JOB01098 with job name YOURID3, enter:

```
CANCEL yourid3(job01098)
```

Cancelling Jobs and Purging Their Output

PURGE/NOPURGE Operands

When you cancel a job, its output is not purged from the system. The NOPURGE operand is a default with the CANCEL command. You can purge a job's output from the system when you cancel that job's execution by specifying the PURGE operand. As with cancelling a job, you can purge the output for all jobs associated with your user ID. To purge a job's output, use the PURGE operand following the job name operand.

Example

To cancel the job with a job name of YOURIDX and purge its associated output, enter:

```
CANCEL youridx PURGE
```

For more information about the CANCEL command, see *z/OS TSO/E Command Reference*.

Chapter 15. Processing the Output of a Batch Job

To display or direct any output that is held by the system, use the OUTPUT command. Held output is output that the system does not immediately place on a print queue. Held output can consist of the job's job control language (JCL) statements, system messages (MSGCLASS), or system output (SYSOUT) data sets.

You can use the OUTPUT command to look at or copy SYSOUT data sets on the hold queue, release them from the hold queue, change their output class or destination, or delete them. On the OUTPUT command, you specify the *job* whose output is to be selected, not the *data set* to be selected.

If your installation uses security labels, you can submit a batch job to run at a security label *that is greater than the security label* you are logged on at. However, you will not be able to process the output created for the submitted job. This is because the output is created with the job's security label, not yours. To view or change the output, you must log on with a security label that is equal to or greater than the security label at which the job ran, and then use the OUTPUT command.

You can view SYSOUT data sets created by your own jobs. By default, this would be jobs corresponding to your user ID. However, if your installation has replaced the default IBM-supplied exit, your jobs could use a different naming convention.

You can also view data sets created by other jobs that have had profiles created authorizing your user ID (through the access list) to look at the data sets. In both cases, the security label you are logged on with must be equal to or greater than the data set's security label.

You can simplify the use of the OUTPUT command by including the NOTIFY operand either on the JOB statement or on the SUBMIT command when you submit a job for batch processing. The system will notify you when the job ends, giving you an opportunity to use the OUTPUT command. MSGCLASS and SYSOUT data sets should be assigned to reserved classes or explicitly held to be available at the terminal.

After you enter the OUTPUT command and the operands you want, the system places your terminal session in OUTPUT mode. At this point, you can enter any of the four subcommands associated with the OUTPUT command to modify a job's held output. The four subcommands are CONTINUE, END, HELP, and SAVE, and are described in “Displaying Output Data Sets with OUTPUT Subcommands” on page 138.

You can also use the OUTPUT command and its subcommands to modify foreground-created output.

Processing the Output of a Batch Job with the OUTPUT Command

Using the OUTPUT command, you can:

- Direct the output from a job to your terminal. The output includes the job's job control language (JCL), system messages (MSGCLASS), and system output (SYSOUT) data sets.
- Direct the output from a job to a specific data set (PRINT operand)
- Control checkpointing of a job (BEGIN, HERE, NEXT operands)
- Control whether the system temporarily pauses after listing a SYSOUT data set (PAUSE/NOPAUSE operands)
- Specify a disposition for held output (KEEP/NOKEEP, HOLD/NOHOLD, DELETE operands)
- Change the output class or classes for a job (NEWCLASS operand)
- Route the output for a job to a remote workstation (DEST operand)

By default, you can use the OUTPUT command to process jobs whose job names begin with your user ID. Access to jobs whose job names do not begin with a valid user ID must be provided by an installation-written exit routine.

© Copyright IBM Corp. 1988, 2024

Displaying Held Output for Specific Jobs

You can display held output for:

- A specific job identified by job name
- A number of jobs
- A specific job identified by a system-supplied job identifier
- A job with a particular output class

To display the held output from the job associated with a specific job name, specify the job name following the OUTPUT command.

Example

To display the output from the job with job name YOURID2, enter:

```
OUTPUT yourid2
```

If there is more than one job associated with a job name, the system prompts you for a specific job identifier.

To display the held output for each job associated with a number of job names, enclose in parentheses the list of job names separated by a comma or blank.

Example

To display the output for the jobs whose job names are YOURID2, YOURID5, YOURIDC, enter:

```
OUTPUT (yourid2,yourid5,youridc)
```

To display the held output for a specific job associated with a job name, specify the job name with the system-supplied job identifier enclosed in parentheses.

Example

To display the held output for the job identified by number JOB02249 with job name YOURID2, enter:

```
OUTPUT yourid2(job02249)
```

To display the held output in a particular output class for the jobs associated with a specific job name, issue the OUTPUT command and specify:

- The job name
- The CLASS operand with the job class or classes to be searched enclosed in parentheses. A class name is a single letter or digit (A-Z or 0-9). If you do not specify the name of a class, all held output for the jobs is available.

Example

To display the held output in output class A, B, or S associated with job name YOURIDX, enter:

```
OUTPUT youridx CLASS(a,b,s)
```

Redirecting Held Output for Specific Jobs

PRINT Operand

You can redirect output for a specific job to:

- A terminal
- A data set

To redirect the held output from the jobs associated with a specific job name to your terminal, issue the OUTPUT command and specify:

- The job name
- The PRINT operand with an asterisk (*) enclosed in parentheses

If you do not specify either a data set name or an asterisk, an asterisk is the default. PRINT is the default value if you omit the PRINT, DELETE, NEWCLASS, DEST, HOLD and NOHOLD operands, which are discussed later in this chapter.

Example

To redirect the held output for the jobs associated with job name YOURID5 to your terminal, enter:

```
OUTPUT yourid5 PRINT(*)
```

To redirect the held output from the jobs associated with a specific job name to a data set, issue the OUTPUT command and specify:

- The job name
- The PRINT operand with the second-level, user-supplied name of the data set to which the output is to go enclosed in parentheses

Example

To redirect the held output for the jobs associated with job name YOURIDX to data set USERID.HLDPRINT.OUTLIST, enter:

```
OUTPUT youridx PRINT(HLDPRINT)
```

In the previous example, the system appends your user ID (YOURID) as the first-level qualifier. It uses the name you specify as the user-specified second-level qualifier, and OUTLIST as the default descriptive qualifier.

Directing Held Output Based on Checkpointing

BEGIN, HERE, and NEXT Operands

A checkpoint is a point at which information about the status of a job and the system can be recorded so that the job step can be restarted later, if necessary. A data set is checkpointed if it was interrupted during printing and never processed to the end of the data during a terminal session. Interruptions that cause a data set to be checkpointed occur when:

- Processing terminates in the middle of printing a data set because of an error or abend condition.
- The attention interrupt key is pressed during the printing of a data set and the CONTINUE NEXT subcommand is entered. The KEEP operand must be present or the data set is deleted.
- The attention interrupt key is pressed during the printing of a data set and the END subcommand is entered.

You can start output processing of held output data sets at the following points:

- At the beginning of each data set, regardless of checkpoints

- At the beginning of each data set, if the data set has not been checkpointed
- At the approximate point of interruption in a checkpointed data set
- After skipping a checkpointed data set

For information on checkpointing a data set, see *z/OS TSO/E Command Reference* and *z/OS MVS JCL User's Guide*.

To start output processing at the beginning of a held output data set that is associated with a job name, regardless of whether the data set has been checkpointed, use the BEGIN operand.

Example

To ensure that the system starts its output processing at the beginning of each data set that contains the held output for each job associated with job name YOURIDX, enter:

```
OUTPUT youridx BEGIN
```

To start output processing at the beginning of a held output data set that is associated with a job name and that has not been checkpointed, use the HERE operand. You can also use the HERE operand to start output processing for the held data sets associated with a job name at the approximate point of interruption, as long as the data set has been checkpointed.

Example

To ensure that the system starts its output processing at the beginning of each data set that is not checkpointed, and that contains the held output for each job associated with job name YOURID5, enter:

```
OUTPUT yourid5 HERE
```

To skip output processing for the held output data sets that are associated with a job name and are checkpointed, use the NEXT operand. Output processing resumes at the beginning of the next uncheckpointed data set. The system deletes the checkpointed data sets it skips, unless you specify the KEEP operand.

Example

To request that the system skip output processing for each checkpointed data set that contains the held output for each job associated with job name YOURID5, enter:

```
OUTPUT yourid5 NEXT
```

Pausing to Process Held Output

PAUSE/NOPAUSE Operand

You can have the system pause after it lists a SYSOUT data set to allow you the opportunity to enter a SAVE or CONTINUE subcommand. Do this by using the PAUSE operand. The NOPAUSE operand, a default with the OUTPUT command, requests that the system not pause after listing each SYSOUT data set.

Example

To have the system pause after it lists each SYSOUT data set associated with the jobs with job name YOURID5, enter:

```
OUTPUT yourid5 PAUSE
```

Pressing the Enter key causes normal processing to continue. To override the PAUSE operand, use the CONTINUE subcommand.

Specifying a Disposition for Held Output

KEEP/NOKEEP, HOLD/NOHOLD, and DELETE Operands

To request that the system keep the held output data sets associated with a specific job name on the output queue after printing them, use the KEEP operand following the job name. After a job is printed, the system releases and deletes the held output data sets. The NOKEEP and NOHOLD operands are the defaults.

Example

To keep the held output data sets associated with the jobs with job name YOURIDX on the output queue after printing them, enter:

```
OUTPUT youridx KEEP
```

When you submit a job to the system and the message class or output class is not a held class, the system does not hold the output data sets. To request that the system hold the output data sets associated with a specific job name so you can display them later, use the HOLD operand following the job name operand.

Example

To hold the output data sets associated with the jobs with job name YOURIDX, enter:

```
OUTPUT youridx HOLD
```

To request that the system delete the output data sets associated with a specific job name, use the DELETE operand following the job name operand.

Example

To delete the output data sets associated with the jobs with job name YOURIDX, enter:

```
OUTPUT youridx DELETE
```

Specifying a New Output Class for Held Output

NEWCLASS Operand

To change the output class for the held output for the jobs associated with a particular job name, use the NEWCLASS operand with the new output class enclosed in parentheses.

Example

To change the output class to R for the held output for the jobs associated with job name YOURIDX, enter:

```
OUTPUT youridx NEWCLASS(r)
```

Routing the Held Output to a Remote Location

DEST Operand

To route the output for the jobs associated with a particular job name to a remote location, use the DEST parameter with the remote node ID enclosed in parentheses.

Example
To route the output from the jobs with job name YOURIDX to remote location PAPUA#NG, enter: `OUTPUT youridx DEST(papua#ng)`

Displaying Output Data Sets with OUTPUT Subcommands

There are four subcommands associated with the OUTPUT command. They are CONTINUE, END, HELP, and SAVE. Use these subcommands to interactively display or direct output data sets. Use OUTPUT subcommands as follows:

- To resume interrupted output operations, use the CONTINUE subcommand
- To terminate OUTPUT, use the END subcommand
- To obtain further information about the OUTPUT command and its associated subcommands, use the HELP subcommand
- To copy a SYSOUT data set to another data set for retrieval by a different method, use the SAVE subcommand

CONTINUE Subcommand

Use the CONTINUE subcommand to resume interrupted output operations. Interruptions causing subcommand mode occur when:

- Processing of a SYSOUT data set completes and the PAUSE operand was specified on the OUTPUT command
- You press the attention interrupt key

Pressing the attention interrupt key purges the input/output buffers for the terminal. Data and system messages in the buffers at this time may be lost. Although the OUTPUT command attempts to back up records to recover the lost information, results are unpredictable due to record length and buffer size. You may see records repeated or you may notice records missing if you attempt to resume processing of a data set at the point of interruption using the HERE operand with the CONTINUE subcommand, or in the next session using the HERE operand on the OUTPUT command.

Using the CONTINUE subcommand, you can specify:

- Where in the output data set you want the system to begin processing (BEGIN, HERE, NEXT operands)
- That you want to skip processing for a checkpointed output data set (NEXT operand)
- Whether you want the system to pause after each SYSOUT data set is listed (PAUSE/NOPAUSE operands)

BEGIN, HERE, and NEXT Operands

To have the system start processing the output data set at the beginning of the data set, regardless of whether it has been checkpointed or not, use the BEGIN operand.

Example
To ensure that the system starts processing the output data set you are currently displaying at the beginning, enter: `CONTINUE BEGIN`

For information on checkpointing a data set, see "Directing Held Output Based on Checkpointing" on page 135 and *z/OS TSO/E Command Reference*.

To have the system start processing the output data set at the beginning of the data set, as long as the data set has not been checkpointed, use the HERE operand. You can also use the HERE operand to have the system start processing the data set at the approximate point of interruption, as long as the data set is checkpointed.

Example

To ensure that the system starts processing the output data set at the beginning, as long as the data set you are displaying is not checkpointed, enter:

```
CONTINUE HERE
```

To have the system skip output processing for a checkpointed output data set, use the NEXT operand. Output processing resumes at the beginning of the next uncheckpointed data set. Note that unless you specify the KEEP operand, the checkpointed data sets the system skips are deleted from the system.

Example

To ensure that the system skip output processing for this data set, as long as the current data set is checkpointed, enter:

```
CONTINUE NEXT
```

PAUSE/NOPAUSE Operands

The PAUSE operand causes the system to pause after it lists each SYSOUT data set. To cause normal processing to continue, press the Enter key.

Example

To have the system pause after listing a SYSOUT data set, enter:

```
CONTINUE PAUSE
```

You can use this operand to override a NOPAUSE specification on the OUTPUT command.

NOPAUSE, a default on the CONTINUE subcommand, specifies that you do not want the system to pause. You can use this operand to override a PAUSE specification on the OUTPUT command.

END Subcommand

To terminate the operation of the OUTPUT command, issue the END subcommand, as follows:

```
END
```

HELP Subcommand

To obtain syntax and function information about all the subcommands of the OUTPUT command, issue the HELP subcommand, as follows:

```
HELP
```

To obtain syntax and function information about a specific subcommand of the OUTPUT command, use the HELP subcommand, followed by the name of the subcommand in question.

Example

To obtain information about the CONTINUE subcommand, enter:

```
HELP CONTINUE
```

SAVE Subcommand

Use the SAVE subcommand to copy the SYSOUT data set from the spool data set to the named data set. The named data set can be any data set that would be valid if used with the PRINT operand. There is no restriction against saving JCL. Using the SAVE command you can specify the name of the data set into which you want to save the output currently being processed.

To use SAVE, you must specify the PAUSE operand on the OUTPUT command. SAVE does not save the entire SYSOUT output of the job; it saves only the data set currently being processed.

To have the system save the output currently being processed into a data set, use the SAVE subcommand followed by the user-specified qualifier of the destination data set's name.

Example

To save the data from the output data set you are currently processing into data set OUTSAVE.OUTLIST, enter:

```
SAVE OUTSAVE
```

Chapter 16. Executing Foreground Commands from a Background Job

There are times when it is not practical to execute a series of commands or a CLIST from your terminal. If a job is going to take an extended period of time to execute or if a large amount of output is to be printed, it is more convenient to execute in the background, that is, independent of the terminal.

TSO/E allows the user to execute a CLIST or a series of commands in the background. To use this function, you *must be* authorized by your installation to use the SUBMIT command or to process JCL.

For a list of restrictions that apply to using commands in the background, see "Command Processing Restrictions in the Background" on page 147.

For a description of commands processing differently in the background, see "Command Processing Differences in the Background" on page 148.

Concurrent Execution of Commands

When executing commands in the background and foreground concurrently, you should be aware that allocation of a data set both in the foreground and the background might not be successful. If a data set has been allocated (foreground or background mode) with a disposition of OLD, MOD, or NEW, it cannot be allocated in the opposite mode by the ALLOCATE command or any other command processor. For example, the ALLOCATE command, issued in the background for a data set in use in the foreground, will issue an error message that the data set is already in use. Any command remaining in the job stream is processed.

If a user's LOGON procedure allocates a data set with a disposition of NEW, MOD, or OLD, a background job specifying the same LOGON procedure will not execute until the user logs off or frees the data set in the foreground. Similarly, if a background job is executing and the user attempts to log on a terminal with the same LOGON procedure, the logon attempt fails.

Output Handling

Output produced by a background job differs from output in the foreground in the following ways:

- Messages producing multiple levels are printed in their entirety. (It is the same as entering a question mark (?) and recovering all levels in the foreground.)
- The allocation of an output data set to the terminal causes that output to be printed after all other output.

If it is necessary to see the output at your terminal rather than waiting to have it printed on the system printer, one of two things can be done:

1. Place the output in a data set
2. Have the output held and use the OUTPUT command to look at it

For the appropriate JCL, see "Writing JCL for Command Execution" on page 144.

Submitting Commands Using the SUBMIT Command

The SUBMIT command is used to submit batch jobs. For more information, refer to "Submitting a Batch Job with the SUBMIT Command" on page 124. There are two techniques for submitting batch jobs using the SUBMIT command.

1. Use the SUBMIT command or subcommand of EDIT. When submitting a batch job in a data set, the data set must be in the same format as a CNTL-type data set. Using the EDIT command to create the data set assures you of the correct data set format. The *first part* of Figure 5 on page 142 illustrates

© Copyright IBM Corp. 1988, 2024

how the EDIT command can be used to create a data set containing commands and how the SUBMIT command is used to submit this same data set as a background TSO/E session.

2. Use the SUBMIT * command from READY mode. The SUBMIT command supports an asterisk (*) for the positional operand value and two keyword operands, END and PAUSE. The keyword operands, END and PAUSE are valid only when (*) is specified and when the issuer is not in EDIT mode. The *second part* of Figure 5 on page 142 illustrates how the SUBMIT command can be used in READY mode.

```
 READY
EDIT examp2.cntl NEW
 INPUT
 00010 LOGON
 00020 PROFILE PREFIX(yourid)
 00030 EDIT a.data NEW EMODE
 00040 5 this is first line
 00050 10 this is second line
 00060 save b.data reuse
 00070 end save
 00080 < null line >
 EDIT
SUBMIT * jobc(a)
 JOB YOURIDA(JOB000347) SUBMITTED
 EDIT
end save
 READY
```

```
 READY
SUBMIT *
 ENTER INPUT JOB STREAM:
LOGON
TIME
LISTBC
 < null line >
 ENTER JOBNAME CHARACTER(S) -
a
 JOB YOURIDA(job04261) SUBMITTED
 READY
```

Figure 5. Submitting Commands in a Batch Job

If you want the SUBMIT command to generate the JCL statements to execute commands in the background, specify LOGON as the first command in the data set. No operands are required on the LOGON command. However, you must specify your user ID if you specified any other operands on the LOGON command.

When the SUBMIT command finds a LOGON command in the SUBMIT job stream before it finds a JOB statement, SUBMIT processing uses the LOGON command to build the JOB and EXEC statements. The job created (if the PROC operand was not on the LOGON command) contains an EXEC statement with PGM=IKJEFT01. Therefore, the job created is the TSO/E terminal monitor program (IKJEFT01) running in the background. All input after the LOGON command is considered to be TSO/E commands.

If your installation uses security labels and security checking, and the SECLABEL operand appears on the LOGON command found in the job stream, SUBMIT processing will include the SECLABEL keyword with its value on the JOB statement it builds. The batch job will then run at the security label given on the JOB statement. If the SECLABEL operand does *not* appear on the LOGON command, the job will run at the security label the user is logged on at, and no SECLABEL keyword will be generated on the JOB statement.

You can submit a batch job that has a security label that is *greater than the security label* you are logged on at, provided you are authorized to use the job's security label. However, during your current TSO/E session, you will not be able to use the OUTPUT command to process the output from the job or the CANCEL command to cancel the job. If the job output is held, you can process it by logging on with a security label that is greater than or equal to the job's security label.

Depending on the security options used at your installation, you may not be able to submit a job at a security label that is *less* than the security label you are currently logged on with.

If you want to charge the job to an account number other than the one you are currently using, you must specify the ACCT operand. MAIL, NOMAIL, NONOTICE, and RECONNECT operands are ignored when you

specify them on a LOGON command executed in the background. Figure 6 on page 143 illustrates how the system integrates your data with the JCL generated by the system from the example in Figure 5 on page 142.

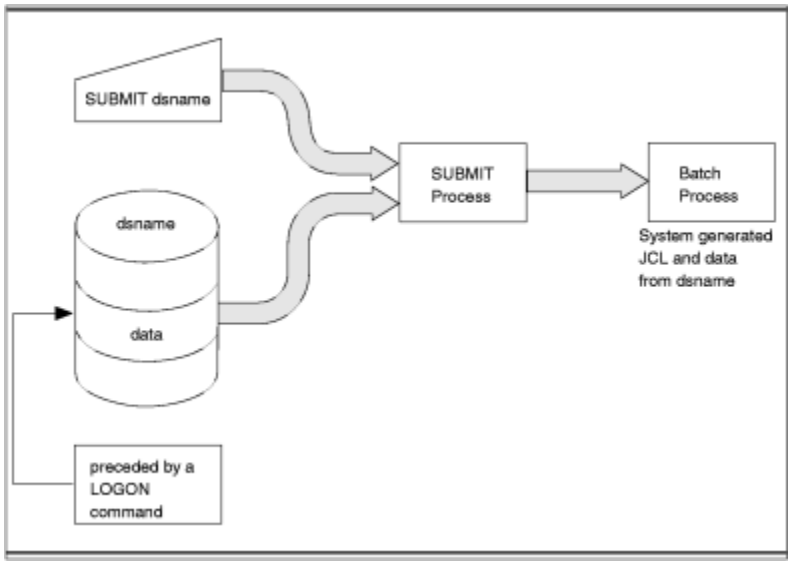


Figure 6. The SUBMIT Process Using System-Generated JCL

Other conditions that apply when submitting jobs in the background are:

- You can precede the LOGON command with JES2 or JES3 JCL statements. If this is done, however, TSO/E does not process any other JCL statements in the job stream, and the system does not generate any JCL statements. If a JCL statement follows the LOGON command, it is executed in the background as an input line.
- You must write the LOGON command on one line in columns 1 through 72 only.
- You can have more than one LOGON command within a single data set. SUBMIT creates the JCL statements for each of them and executes them accordingly.
- You cannot use the characters */ in columns one and two in a data set where SUBMIT-generated JCL is to be used. SUBMIT designates these characters as a delimiter. If you must use these characters in columns one and two, you have to provide your own JCL.
- Use only the PROC operand on a submitted LOGON command if the procedure is available in the SYS1.PROCLIB.
- The PROC specified on a submitted LOGON command must not contain the ddnames SYSTSIN or SYSTSPRT. SUBMIT always generates these ddnames.

For example, if you submit a data set containing the commands LOGON and LISTCAT, SUBMIT enters the following background job:

```
//YOURIDA    JOB    ACCT.INFO.,
//                  YOURID
//                  NOTIFY=YOURID
//                  MSGLEVEL=(1,1)
//*******************************************************
//*  THE FOLLOWING LOGON COMMAND WAS FOUND IN SUBMIT'S   *
//*  INPUT DATA SET(S) AND WAS USED TO GENERATE THE JCL  *
//*  TO EXECUTE TSO/E COMMANDS IN THE BACKGROUND:        *
//*                                                      *
//*  LOGON                                               *
//*                                                      *
//*******************************************************
//CBSTEP     EXEC PGM=IKJEFT01,DYNAMNBR=30
//SYSTSPRT   DD   SYSOUT=A  **OUTPUT FROM COMMANDS IN
                            BACKGROUND**
//SYSTSIN    DD   DATA,DLM='*/'  **INPUT COMMANDS**
  LISTCAT
*/
```

If the above SUBMIT-generated JCL statements are not sufficient, you can insert your own JCL. Do not include a LOGON command if this is done. Figure 7 on page 144 illustrates the SUBMIT process if you create your own JCL.

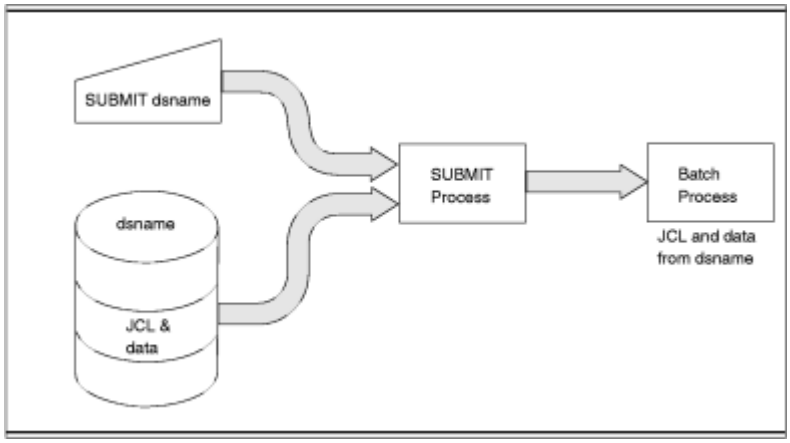


Figure 7. The SUBMIT Process With User-Created JCL Statements

Submitting Jobs in TSO Batch

When submitting a background job (that is, a job in TSO batch) that contains foreground commands, you must use at least one of each type of the JCL statements shown in Figure 8 on page 144.

Figure 8 on page 144 illustrates the JCL statements required for input and the appropriate placement of the data statements.

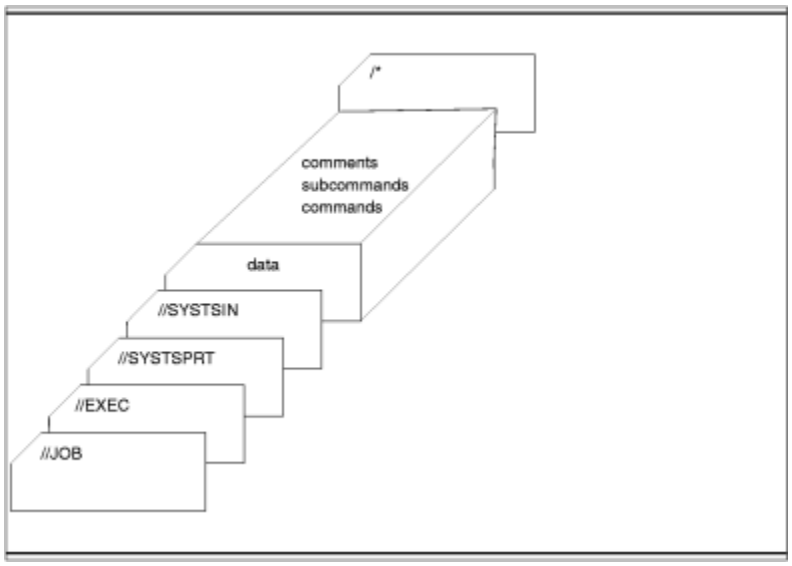


Figure 8. JCL Setup for Processing Commands in the Background

Writing JCL for Command Execution

The following JCL statements are required for executing commands in the background. Other statements can be used, but are not required. For a complete description of JCL statements, see *z/OS MVS JCL User's Guide*. *z/OS TSO/E Customization*, Appendix A, contains additional information about processing TSO/E commands in the background.

JOB Statement

The JOB statement is the first JCL statement in a batch job. It marks the beginning of a job and, when jobs are stacked in the input stream, marks the end of the control statements for the preceding job. This statement consists of four fields. The format is:

```
//jobname JOB operands comments
```

where:

jobname
is required and is used to identify the job to the system.

JOB
is the JCL term that identifies the type of statement.

operands
specifies the specific processing options the system uses when processing this job.

comments
specifies optional user-supplied information pertaining to the function of this statement.

EXEC Statement

The program needed to execute TSO/E commands from the background is a terminal monitor program (TMP), which may be one of the following: IKJEFT01, IKJEFT1A, or IKJEFT1B. The EXEC (execute) statement is used to execute program IKJEFT01 or the alternate programs IKJEFT1A and IKJEFT1B. The format is:

```
//stepname EXEC PGM=IKJEFT01,DYNAMNBR=nn,PARM='command'
or
//stepname EXEC PGM=IKJEFT01,DYNAMBR=nn,PARM='command'
```

Note: The TSO/E terminal monitor program must not run as a V=R program. Executing the TMP as V=R will cause unpredictable results.

Note: If the caller wishes to pass a command string longer than 100 characters to IKJEFT01, you should use the PARMDD= keyword of the EXEC statement of JCL, as shown above.

stepname
is optional and can be used as a step identifier in programs consisting of more than one step.

EXEC
is the JCL term that identifies this type of statement.

PGM=
specifies the module being executed. In addition to IKJEFT01, there are two other entry points available for background execution that provide additional return code and abend support.

Note: If the STEP parameter is coded in an ABEND macro, the job will abnormally terminate with the system or user abend specified. TSO/E cannot recovery normally from an ABEND with the STEP parameter because retry processing is not allowed.

The differences among the three entry points are:

- PGM=IKJEFT01
  - When a command completes with a non-zero return code, the program goes to the next command. When a command abends, the step ends with a condition code of 12 (X'C').
- PGM=IKJEFT1A
  - If a command or program being processed by IKJEFT1A ends with a system abend, IKJEFT1A causes the job step to terminate with a X'04C' system completion code. IKJEFT1A also returns to the caller the completion code from the command or program in register 15.
  - If a command or program being processed by IKJEFT1A ends with a user abend, IKJEFT1A saves the completion code in register 15 and then terminates.

  - If a command, program or REXX exec being processed by IKJEFT1A returns a non-zero return code to IKJEFT1A, IKJEFT1A saves this return code in register 15 and then terminates. Non-zero return codes to IKJEFT1A from CLISTs will not affect the contents of register 15 and the TMP will continue processing.
  - For a non-zero return code or an abend from a command or program that was not given control directly by IKJEFT1A, no return code is saved in register 15, and IKJEFT1A does not terminate.

  Note: When a command is invoked in a CLIST, that command is actually processed by the TMP (Terminal Monitor Program). From that command, termination of IKJEFT1A with a non zero return code and flushing of the remainder of the CLIST occur. This is different from the case when a command is invoked directly by a REXX exec.

- PGM=IKJEFT1B
  - If a command or program being processed by IKJEFT1B ends with a system or user abend, IKJEFT1B causes the job step to terminate with a X'04C' system completion code. IKJEFT1B also returns to the caller the completion code from the command or program in register 15.
  - If a command, program or REXX exec being processed by IKJEFT1B returns a non-zero return code to IKJEFT1B, IKJEFT1B saves this return code in register 15 and then terminates. Non-zero return codes to IKJEFT1B from CLISTs do not affect the contents of register 15 and the TMP continues processing.
  - For a non-zero return code or abend completion code from a program or command that was not given control by IKJEFT1B, no return code is saved in register 15, and IKJEFT1B does not terminate.

  See *z/OS TSO/E Customization* for information on abend and non-zero return code processing by these alternate programs, as well as conditional disposition processing for data sets.

DYNAMNBR=
indicates the number of system resources the system should hold in anticipation of reuse. For information on the specific number of allocations you can specify, see *z/OS MVS JCL Reference*.

PARM=
is optional and can be used to supply the first (or only) command to be executed. This is used most often when you only execute one command in the step.

PARMDD=ddname
is optional and can be used instead of PARM= to supply the first (or only) command to be executed. Use PARMDD specifying the ddname of a data set containing the command parmstring to be executed if the command parmstring is more than 100 characters in length.

SYSTSPRT DD Statement

The SYSTSPRT DD statement is used to control the output from your background job. By specifying different operands on this statement, you can have the output listed on a system printer, placed in a specified data set for later use, or held in a work data set, so you can look at it using the OUTPUT command.

If you want to see your output as soon as the job has executed, hold the output and specify the NOTIFY=user ID operand on the JOB statement. NOTIFY causes a message to be displayed at your terminal when the job completes. For example, assume that you specified NOTIFY=your user ID on the JOB statement. To hold the output, enter:

```
//SYSTSPRT  DD  SYSOUT=H
```

where H is a held class and MSGCLASS=H is also specified on the JOB statement.

You can also hold output by specifying:

```
//SYSTSPRT  DD  SYSOUT=N,HOLD=YES
```

where N is not a held class.

You might not always want to hold a job's output. To not hold a job's output, specify the SYSOUT operand with the non-held output class value. For example,

```
//SYSTSPRT  DD  SYSOUT=N
```

where N is the installation-defined class for output that is not held.

You can write a job's output to a specified data set. For example, to write a job's output to new data set PREFIX.JOBOUT.DATA, code the following:

```
//SYSTSPRT  DD  DSN=PREFIX.JOBOUT.DATA,DISP=NEW,SPACE=(TRK,5,5),
         UNIT=SYSDA
```

To write a job's output to old data set PREFIX.OLDOUT.DATA, specify the following:

```
//SYSTSPRT  DD  DSN=PREFIX.OLDOUT.DATA,DISP=OLD
```

SYSTSIN DD Statement

The SYSTSIN DD statement is used to specify that the data to follow consists of executable commands and/or subcommands. For example, to indicate to the system that all data following this statement is to be used as input, until the system encounters an input delimiter, such as the characters /* or the DLM operand, specify:

```
//SYSTSIN  DD  *
```

If any of the input statements start with the characters **//**, use the DD DATA statement instead.

To indicate to the system that all data following this statement is to be used as input, including statements that start with the characters //, until an input delimiter (/* or DLM) is found, specify:

```
//SYSTSIN  DD  DATA
```

To indicate to the system that all the input data can be found in data set PREFIX.INPUT.DATA, specify:

```
//SYSTSIN  DD  DSNAME=PREFIX.INPUT.DATA
```

The SYSTSIN and SYSTSPRT DD statements can refer to a sequential data set or a member of a partitioned data set.

It is recommended that the SYSTSIN DD be defined as a fixed block format data set, with an LRECL of 80.

If SYSTSIN is a fixed length data set (FB), the last 8 bytes of the record will be treated as a sequence number and ignored.

If SYSTSIN is a fixed length data set with ASA control characters (FBA), the first byte of the record will be treated as a carriage control character and ignored.

If SYSTSIN is a variable length data set (VB), the first 8 bytes of the record will be treated as a sequence number and ignored.

If SYSTSIN is a variable length data set with ASA control characters (VBA), the first 9 bytes of the record will be treated as a sequence number, followed by a carriage control character and ignored.

You cannot refer to concatenated data sets on the SYSTSIN DD statement. Each command or subcommand must begin on a separate statement.

Command Processing Restrictions in the Background

General Restrictions

These restrictions always apply to the execution of commands processed in the background:

- You cannot use the OPERATOR and TERMINAL commands because they are not supported. You get an error if you try to issue them.
- You cannot specify USER(*) on the SEND command in the background. A user ID is required. USER(*) causes an error message to be issued.
- Specifying RECOVER with the EDIT command has no affect. The EDIT CLIST workfile recovery function is not supported.
- You cannot use the CLIST statements READ and TERMIN because they are not supported.
- The symbolic variable &SYSPROC returns the current procedure name (either LOGON procedure name, Started Task procedure name, or 'INIT' for a batch job). For more information, see *z/OS TSO/E CLISTs*.

Non-RACF Restrictions

The following restrictions apply only to the execution of commands processed in the background when RACF is not installed on your system.

- The IBM-supplied installation exit rejects the CANCEL, RECEIVE, and OUTPUT commands because no user ID is available in the background.
- You must specify operands on the STATUS command. The system issues an error message if none is supplied.
- A PROFILE command with a PREFIX (user_id) operand is required if you want a user ID to be prefixed to all data-set-names, or if something other than a null value is required for the CLIST symbolic variable, &SYSPREF.
- The CLIST symbolic variable &SYSUID has a null value.
- When a data set containing a job to be submitted in the background does not contain a JOB statement, SUBMIT generates one. It attempts to construct the job name using the value on the USER operand. If there is no value with the USER operand, the system issues a warning message and uses the default job name SUBMIT JB.
- When generating a JOB statement, SUBMIT does not insert a NOTIFY=user ID operand.

Note: In a system with RACF installed, there are fewer restrictions for users that are both RACF and TSO/E defined. When RACF is installed, TSO/E can establish profile attributes for a background job, provided the JOB statement contains a value for the USER operand. TSO/E accesses the UADS entry associated with the user identified on the USER operand on the JOB statement.

For more information about processing considerations for RACF and non-RACF systems, see the PROFILE command in *z/OS TSO/E Command Reference*.

Command Processing Differences in the Background

When executed in the background, some TSO/E commands process differently from when they are executed in the foreground. The commands listed here show only the differences between how they process in the background as opposed to how they process in the foreground. The command syntax is shown only where operands have been added. For a complete description of each of these commands, see *z/OS TSO/E Command Reference*.

ALLOCATE Command

The Default UNIT Operand Value

If the ALLOCATE command is used without UNIT operand, the default UNIT operand value depends on whether the ALLOCATE command is executed in the foreground or in the background.

Processing in the Foreground

The ALLOCATE command uses the default UNIT operand value from the protected step control block (PSCB).

During LOGON processing this value is taken from either the user attribute data set SYS1.UADS or the security system being used (if SYS1.UADS is not being used) and put in the PSCB (PSCBGPNM field). See the ACCOUNT command and its UNIT operand in *z/OS TSO/E System Programming Command Reference* on how to specify a default device type in SYS1.UADS.

Processing in the Background

If the Terminal Monitor Program (TMP) IKJEFT01 (or IKJEFT1A or IKJEFT1B) is executed in the background and the ALLOCATE command without UNIT operand is issued, the default value from the PSCB is nulls, because LOGON is not executed.

Search Order for UNIT Operand Value

The UNIT operand value is searched for in the following sequence:

1. The catalog for this data set, provided the VOLUME operand of the ALLOCATE command is not being used.
2. The protected step control block (PSCB).
3. The UNIT keyword on the ALLOCxx member of SYS1.PARMLIB.

   If there is a UNIT keyword, and Storage Management Subsystem (SMS) is installed and active and has a default UNIT in the SMS control data set (SCDS), the latter one is used instead.
4. The MVS system default SYSALLDA, which contains all DASD defined to the system.

CALL Command

Service aids, utilities, and other programs obtaining their input from an allocated file such as SYSIN, must have the input in a created data set or a job stream data set that is created with a DD * or a DD DATA statement. After the data set has been created and allocated, the CALL command can be used to execute the program that accesses the SYSIN data. Figure 9 on page 149 illustrates the allocation and creation of input data sets.

```
//examp1   exec    pgm=ikjeft01,dynamnbr=20
//systsprt dd      sysout=a
//systsin  dd      *
  PROFILE PREFIX(yourid)
  ALLOCATE FILE(sysprint) DATASET(*)
  ALLOCATE FILE(sysin) ALTFILE(inputdd)
  CALL prog1
  ALLOCATE FILE(sysin) ALTFILE(inputdd2) REUSE
  CALL prog2
  FREE ALL
//inputdd  dd     *
  **input to prog1**
//inputdd2 dd     *
  **input to prog2**
/*
```

Figure 9. Allocating and Creating Input Data Sets

In the preceding example, the second allocate statement allocates the SYSIN data for PROG1, and specifies that the data is defined by the DD statement named INPUTDD. The third allocate statement allocates the SYSIN data for PROG2, and specifies that the data is defined by the DD statement named INPUTDD2. DD statements INPUTDD and INPUTDD2 use in-stream data as input.

Note: Allocating the input file to a terminal results in an I/O error message. Termination occurs when the program tries to get input from the terminal.

EDIT Command

Although EDIT performs the same in the foreground and background, you must be careful when inserting blank lines into a data set because EDIT interprets a blank line as a null line. This causes the system to switch from input mode to edit mode. To insert blank lines into a data set, you can:

- Insert a character string that you do not use anywhere else in the data set. Then, before ending the EDIT session, issue a global change, changing every occurrence of the character string to blanks.
- Specify the EMODE operand on the EDIT command. Each new line is preceded by a line number, wherever line numbering is allowed.

For an example of how to insert blank lines into a data set using line mode EDIT, see Figure 19 on page 197.

LOGOFF Command

When the LOGOFF command is executed in the background, your TSO/E session is terminated normally. Any remaining commands in the input stream are ignored.

PROFILE Command

The following differences should be noted for foreground/background processing:

- Changes made while processing in the foreground are saved from session to session.
- Changes made while processing in the background are not saved.

RECEIVE Command

You can use the RECEIVE command in a background job to receive a message or data set. However, if your installation uses security labels, there will be security constraints placed on the use of the RECEIVE command. These constraints are explained in “Security Considerations for Sending and Receiving Data Sets” on page 94. To use RECEIVE in the background, you must specify the RECEIVE command and the responses to any prompts that you would anticipate when using RECEIVE in the foreground. Supply the responses in the same order as they would appear in the foreground. The commands are executed as if PROFILE NOPROMPT was entered.

The data set or message that you receive will be prefixed by the user ID that you supply on the JOB statement. To use an alternate prefix as the data set name, include the following command at the beginning of the SYSTSIN data stream:

```
PROFILE PREFIX(prefix)
```

Be sure that any alternate prefix you use is a valid catalog alias.

Messages are written to the data set defined in the SYSTSPRT DD statement.

Figure 10 on page 150 shows a batch job for receiving a data set in the background.

```
//JOBNAME  JOB     USER=USERID,PASSWORD=PASSWD
//STEP1    EXEC    PGM=IKJEFT01,REGION=512K
//SYSTSPRT DD      SYSOUT=A
//SYSTSIN  DD      *
  RECEIVE
  RESTORE
  **responses to anticipated prompts**
  r
/*
```

Figure 10. Receiving a Data Set in the Background

Figure 11 on page 150 shows a batch job for receiving a message in the background.

```
//JOBNAME  JOB     USER=USERID,PASSWORD=PASSWD
//STEP1    EXEC    PGM=IKJEFT01,REGION=512K
//SYSTSPRT DD      SYSOUT=A
//SYSTSIN  DD      *
  RECEIVE
/*
```

Figure 11. Receiving a Message in the Background

Handling Error Conditions

The return code from the job step that executes commands in the background is that of the last command executed.

For information on specific abend codes, see *z/OS MVS System Codes*. Abends occurring in the background will be handled in the following manner:

XXX

ABEND code '*xxx*' occurs when a command or the TMP abends. In this situation, the session terminates. If a SYSUDUMP or SYSABEND was allocated either in the JCL or with the ALLOCATE command, a dump occurs and the remainder of the commands in SYSTSIN are ignored.

Part 4. Changing the Way You Use TSO/E

After you are familiar with your terminal, you might want to change certain terminal characteristics. You can change the way you use your terminal with the following:

PROFILE command
Controls the user profile, which defines how you interact with TSO/E and with other users.

TERMINAL command
Defines the dimensions of your display screen.

ISPF/PDF Option 0
Defines terminal characteristics to ISPF/PDF, such as LOG/LIST data set defaults and PF key definitions.

Session Manager
Allows you to keep a record of your terminal session.

Enhanced Connectivity Facilities
Allows communication between IBM PCs and IBM host computers.

© Copyright IBM Corp. 1988, 2024

Chapter 17. Customizing Your Terminal Session

Changing Your User Profile with the PROFILE Command

Your user profile defines how you want the system to respond to information sent to or from your terminal. When you began to use TSO/E, an authorized system programmer created your user ID and your user profile. The typical user profile is defined by default values of the PROFILE command's operands.

You can use the PROFILE command to display or change your user profile.

Example

To display your user profile, enter the PROFILE command after the READY mode message:

```
PROFILE
```

You then see something that might look like the following:

```
CHAR(0) LINE(0) PROMPT INTERCOM NOPAUSE NOMSGID NOMODE
NOWTPMSG NORECOVER PREFIX(USERID) PLANGUAGE(JPN) SLANGUAGE(ENU)
DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS
TERMINAL
```

To change your user profile, enter the PROFILE command and the operands you want to change:

```
PROFILE changed operands
```

The changes made while processing in the foreground usually remain in effect from one session to another. If the changes you requested do not remain in effect, your installation might have a LOGON pre-prompt exit that prevents the system from saving changes to your user profile. Changes made while processing in the background are not saved after the terminal session.

Using the PROFILE command, you can specify:

- On terminals without delete keys, the characters that delete a single character or the remainder of a line (CHAR operand, LIST operand)
- Whether you want the system to prompt you for information (PROMPT/NOPROMPT operand)
- Whether you want to receive messages from other users (INTERCOM/NOINTERCOM operand)
- Whether you want to be able to obtain information about messages issued to your terminal while you execute a CLIST (PAUSE/NOPAUSE operand)
- Whether you want to see the message numbers of messages displayed at your terminal (MSGID/NOMSGID operand)
- Whether you want to receive mode messages at your terminal (MODE/NOMODE operand)
- Whether you want to see at your terminal messages issued to you by a program (write-to-programmer messages) (WTPMSG/NOWTPMSG operand)
- Whether you want the EDIT recovery function in effect (RECOVER/NORECOVER operand)
- A user ID or character string to be used as the first qualifier of all non-fully-qualified data set names (PREFIX operand).
- Primary and secondary languages (PLANGUAGE and SLANGUAGE operands) to be used in displaying translated messages, help information, and the TRANSMIT full-screen panel.

In the following examples, you will see either no operands or one operand entered with each example of the PROFILE command. However, you can enter any number of non-conflicting operands on a PROFILE command.

© Copyright IBM Corp. 1988, 2024

If you enter an incorrect character on the PROFILE command, an error message informs you which character is incorrect and the user profile is not changed.

Specifying a Deletion Character and a Line Deletion Character

CHAR/NOCHAR Operand

The CHAR operand is used on certain terminals to identify a deletion character that deletes a single character from the screen. Some terminals have a built-in delete key on the keyboard. For example, a 3270 display terminal has a delete key marked with **DEL**. When a terminal has a delete key, the CHAR operand might be negated and appear in a user profile as CHAR(0).

On terminals without a delete key, the backspace key is the default for a deletion character and appears in a user profile as CHAR(BS). You can change a deletion character with the PROFILE command by specifying the CHAR operand and changing the character in parentheses after the CHAR operand.

Example

To change a deletion character to **$**, enter after the READY mode message:

```
PROFILE CHAR($)
```

To specify that no character be used as a character deletion key, enter the NOCHAR operand.

LINE/NOLINE Operand

The LINE operand is used on certain terminals to identify the deletion character that deletes the remainder of a line from the screen. Some terminals have a built-in delete line key on the keyboard. For example, a 3270 display terminal had a delete line key marked with **ERASE EOF**. When a terminal has a delete line key, the LINE operand might be negated and appear in a user profile as LINE(0).

On terminals without a delete line key, the attention key is the default for a line deletion character and appears in a user profile as LINE(ATTN). You can change a line deletion character with the PROFILE command by specifying the LINE operand and changing the character in parentheses after the LINE operand.

Example

To change a line deletion key to **#**, enter after the READY mode message:

```
PROFILE LINE(#)
```

To specify that the terminal not have a line deletion character, enter the NOLINE operand.

When specifying deletion characters, avoid using a blank, tab, comma, asterisk, or parentheses because these characters are used when entering commands. Also avoid using alphabetic or other commonly-used characters.

Requesting to be Prompted by the System

PROMPT/NOPROMPT Operand

When you enter incorrect information with a command or when the system requires further information from you, the system issues a message to prompt you for the information. To control whether you receive these system messages, use the PROMPT/NOPROMPT operands. If you do not want to receive the system messages, use the NOPROMPT operand. The PROMPT operand allows you to receive these messages at the terminal and is the default with the PROFILE command.

Example

To request that the system not prompt you for missing or not valid information, enter after the READY mode message:

```
PROFILE NOPROMPT
```

Receiving Messages from Other Users

INTERCOM/NOINTERCOM Operand

The INTERCOM/NOINTERCOM operands control whether you receive messages from other users. INTERCOM allows you to receive those messages and is the default value.

Example

If you do not want to receive messages from other users, enter after the READY mode message:

```
PROFILE NOINTERCOM
```

Obtaining Additional Diagnostic Information

PAUSE/NOPAUSE Operand

To obtain additional levels of information about messages you receive while executing a CLIST, use the PAUSE operand. When the system issues a message with additional levels of information, it displays the word **PAUSE**, and waits for you to either enter a question mark (?) or press the Enter key. When you enter a question mark, the system displays the next level of information associated with the message. When you press the Enter key, the system resumes processing where it left off.

Example

To be able to obtain additional levels of information about messages you receive while executing a CLIST, enter:

```
PROFILE PAUSE
```

If you do not want to be able to obtain additional information about messages you receive while executing a CLIST, use the NOPAUSE operand. NOPAUSE is the default value.

Displaying Message IDs with Messages

MSGID/NOMSGID Operand

The MSGID/NOMSGID operands determine whether you see a message identifier with a message. A message identifier allows you to look up additional information about the message in *z/OS TSO/E Messages*.

A message identifier appears before a message and might look like:

```
IKJ56700A  ENTER USERID -
```

Example

To see the message identifier on messages that appear at your terminal, enter the MSGID operand after the READY mode message:

```
PROFILE MSGID
```

NOMSGID is the default value.

Receiving Mode Messages

MODE/NOMODE Operand

The MODE/NOMODE operands control whether you receive mode messages.

Example

If you do not want to receive mode messages, enter after the READY mode message:

```
PROFILE NOMODE
```

Receiving Write-to-Programmer Messages

WTPMSG/NOWTPMSG Operand

Some programs, as they run, issue messages to the programmer. The default value for a user profile is NOWTPMSG, which means that you do not see write-to-programmer messages displayed on your terminal. To display all write-to-programmer messages, change your user profile to the WTPMSG operand. Setting this operand to WTPMSG may provide additional error messages if system ABENDs occur during your TSO/E session.

Example

To ensure that you receive all write-to-programmer messages, enter after the READY mode message:

```
PROFILE WTPMSG
```

Activating the Edit Recovery Function

RECOVER/NORECOVER Operand

The RECOVER/NORECOVER operands control whether the EDIT recovery function is in effect. For more information about the EDIT recovery function, see “Recovering an EDIT Work File” on page 201.

Example

If you want the EDIT recovery function to be in effect, enter after the READY mode message:

```
PROFILE RECOVER
```

NORECOVER is the default value.

Specifying a Data Set Name Prefix

PREFIX/NOPREFIX Operand

The PREFIX operand determines the first qualifier to be added to all data sets that are not fully qualified. To request the system use a specific user ID or other valid string as the first qualifier of data sets that are not fully qualified, use the PREFIX operand followed by the user ID or string enclosed in parentheses.

Example

To use NEWID as the first-level qualifier of all non-fully-qualified data sets, enter after the READY mode message:

```
PROFILE PREFIX(newid)
```

Then when you specify a data set without quotes, such as TEST.DATA, the system recognizes it as NEWID.TEST.DATA.

If you do not want to append a prefix to non-fully-qualified data sets, enter:

```
PROFILE NOPREFIX
```

The default prefix for foreground processing is your user ID. NOPREFIX is the default for background processing.

Note: When RACF or an Other Equipment Manufacturer (OEM) security server is installed, the default is the user ID. Otherwise, the default is no prefixing of data set names.

Specifying Languages for Message and Help Text Displays

PLANGUAGE Operand

The PLANGUAGE operand identifies the primary language to be used in displaying translated messages, help information, and the TRANSMIT full-screen panel. To change the primary language, use the PLANGUAGE operand followed by 3-character language code or an installation-defined name for the language you want to use. You must enclose the language value in parentheses. The primary language that you specify must be active on your system. If the primary language fails, the secondary language is used; if both fail, U. S. English is used.

See your system administrator for a list of valid language codes and installation-defined names.

Example

To specify Japanese as the primary language, enter after the READY mode message:

```
PROFILE PLANGUAGE(JPN)
```

SLANGUAGE Operand

The SLANGUAGE operand identifies the secondary language to be used in displaying translated messages, help information, and the TRANSMIT full-screen panel should the primary language fail. To change the secondary language, use the SLANGUAGE operand followed by a 3-character language code or an installation-defined name for the language you want to use. You must enclose the language in parentheses. The secondary language that you specify must be active on your system.

See your system administrator for a list of valid language codes and installation-defined names.

Example

To specify U.S. English as the secondary language, enter after the READY mode message:

```
PROFILE SLANGUAGE(ENU)
```

Displaying Your Current User Profile

LIST Operand

To change your profile and list the changed user profile, enter the PROFILE command with changed operands and the LIST operand. The system first changes your user profile according to the other operands you specified, and then lists the changed user profile.

Example

To change your user profile to MSGID and then list the changed profile, enter after the READY mode message:

```
PROFILE MSGID LIST
```

You then see a changed profile that might look like:

```
CHAR(0) LINE(0) PROMPT INTERCOM NOPAUSE   MSGID NOMODE
NOWTPMSG NORECOVER PREFIX(USERID) PLANGUAGE(JPN) SLANGUAGE(ENU)
DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS
TERMINAL
```

Note: When you issue the PROFILE command with no operands, the LIST operand is the default, and you see a list of your current profile.

For more information about the PROFILE command, see *z/OS TSO/E Command Reference*.

Changing the Dimensions of Your Display Screen

The TERMINAL command controls the dimensions of your display screen.

Most installations provide a default TERMINAL command through the logon procedure (PROC) in the LOGON command. The default TERMINAL command is set up for your terminal screen size.

You can use the TERMINAL command to define or alter the display characteristics of your terminal, such as:

- The maximum number of characters allowed per line on a terminal other than an IBM 3270 (LINESIZE operand).
- The screen dimensions of an IBM 3270 (SCRSIZE operand).

The options specified with a TERMINAL command remain in effect until the end of the session or until you issue another TERMINAL command.

If you terminate a TSO/E session and begin a new session by entering the LOGON command, (instead of a LOGOFF command followed by a LOGON command) the terminal characteristics defined in the earlier session are used during the subsequent session.

If a line disconnection interrupts your session, and you log on again using the LOGON command with the RECONNECT option, the terminal characteristics defined earlier in the session are not saved. You must redefine your terminal characteristics because all records for defined data are lost as a result of a line disconnection.

The TERMINAL command is not allowed as a TSO/E command in the background. For a description of foreground and background execution, see Chapter 16, “Executing Foreground Commands from a Background Job,” on page 141.

In the following examples, you will see only one operand entered with each example of the TERMINAL command. You may enter any number of non-conflicting operands on a TERMINAL command.

Specifying the Maximum Characters Per Line

LINESIZE Operand

If your terminal is not an IBM 3270 or equivalent, you can use the LINESIZE operand to specify the maximum number of characters allowed on one line of your terminal.

Example

To specify 40 as the maximum number of characters the system can display on one line of your terminal, enter after the READY mode message:

```
TERMINAL LINESIZE(40)
```

If you enter LINESIZE(80) and try to view a data set with its record format attribute (RECFM) set to U, specifying records of undefined length, the line is truncated at the 79th character. The truncated byte (80) is reserved for an attribute character.

REXX processing formats the output of the REXX SAY instruction and tracing output to one less than the value of the LINESIZE setting. The REXX LINESIZE() function also returns this amount.

Specifying Your Terminal's Screen Size

SCRSIZE Operand

If your terminal is an IBM 3270, you can use the SCRSIZE operand to specify the maximum number of horizontal rows of characters per screen and the maximum number of characters per row.

Example

To specify a maximum of 32 rows of data where each row is a maximum of 80 characters, enter after the READY mode message:

```
TERMINAL SCRSIZE(32,80)
                 /   \
             rows    maximum characters
```

For more information about the TERMINAL command, see *z/OS TSO/E Command Reference*.

Using ISPF/PDF to Customize Your Terminal Session

If you use TSO/E with ISPF/PDF, you can customize terminal characteristics as defined to ISPF/PDF by selecting the ISPF PARMS option (option 0).

```
------------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION ===> 0
                                                              USERID   - YOURID
    0  ISPF PARMS  - Specify terminal and user parameters     TIME     - 12:47
    1  BROWSE      - Display source data or output listings   TERMINAL - 3277
    2  EDIT        - Create or change source data             PF KEYS  - 12
    3  UTILITIES   - Perform utility functions
    4  FOREGROUND  - Invoke language processors in foreground
    5  BATCH       - Submit job for language processing
    6  COMMAND     - Enter TSO command or CLIST
    7  DIALOG TEST - Perform dialog testing
    8  LM UTILITIES- Perform library administrator utility functions
    9  IBM PRODUCTS- Additional IBM program development products
    C  CHANGES     - Display summary of changes for this release
    T  TUTORIAL    - Display information about ISPF/PDF
    X  EXIT        - Terminate ISPF using log and list defaults

Enter END command to terminate ISPF.
```

Some of the characteristics you can customize are:

1. Terminal characteristics for IBM terminals, such as number of program function (PF) keys and whether the input fields are padded with nulls or blanks.
2. ISPF LOG and LIST data set defaults, such as default processing options and a default job statement.
3. PF key defaults.
4. Screen display characteristics, such as whether the COMMAND/OPTION line appears at the top or bottom of the screen.
5. List data set characteristics, such as the record format and length.

For more information about ISPF/PDF and about the ISPF PARMS option, see *z/OS ISPF User's Guide Vol I*.

Chapter 18. Session Manager

What is Session Manager?

TSO/E Session Manager keeps a complete journal of everything that happens during your terminal session while you are in line mode TSO/E. It records everything you type in and everything the system displays. Any time during your terminal session, you can look at work you did in the beginning, middle, or at the end of your session. Session Manager also lets you print a copy of this information.

The session journal can be compared to a long sheet of paper that fills up with information as you do work. PF keys are set up to perform certain functions for you. The specific PF key and their functions are discussed in a later part.

By using Session Manager, you can:

- Keep a record of your interaction with TSO/E.
- Re-issue previous commands without retyping them.
- Find words or character strings previously displayed in your terminal session.
- Print a hardcopy of your TSO/E interaction.

For example, you might want to use Session Manager when you use the TEST command to debug programs. You can then keep track of which data areas you have viewed, and print the output from the TEST command.

To use Session Manager, you need access to a LOGON procedure that recognizes Session Manager.

This chapter describes how to use the default version of Session Manager and how you can change the Session Manager environment.

Using Session Manager

In the default version of Session Manager, the display screen is predefined to contain five windows, the PF keys are set to default values, and Session Manager streams (records of session input and output) are set to specific requirements.

The Display Screen

The Session Manager display screen appears when you press the Enter key after logging on. This screen is divided into several areas called *windows*. You use these windows to enter, look at, and change the work in your session journal.

Windows
Definition

MAIN
The large portion of the display screen above the numbered line is the *MAIN window.* This window lets you see everything you type (commands and input to TSO/E) and everything the system displays (output from commands, messages). In effect, it lets you see your *session journal*. You can also enter information in this window.

CURRENT
Just below the numbered line is the *CURRENT window.* When you first log on, this window lets you see the last two lines in your session journal. As you do work and more output is placed in your session journal, the CURRENT window changes to reflect the last two lines of output.

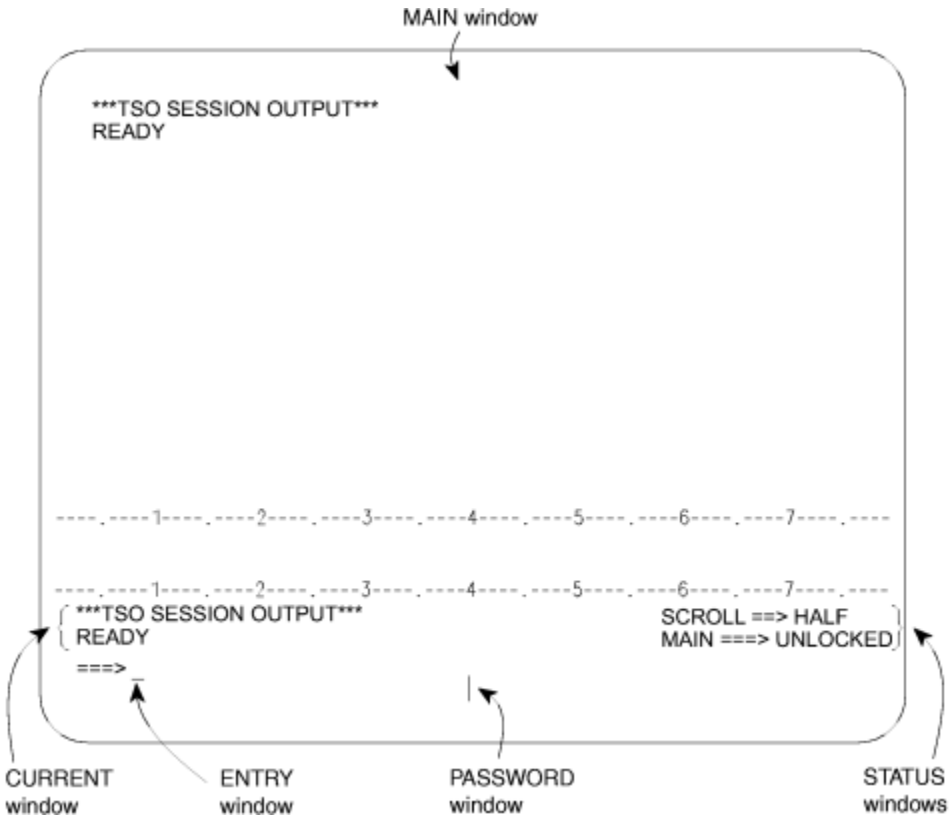
© Copyright IBM Corp. 1988, 2024

STATUS
The *STATUS windows* for the MAIN window are located in the lower right-hand corner of the display screen. The top window shows the *scroll amount* (labelled SCROLL===> HALF) and the bottom window indicates whether the MAIN window is *locked or unlocked.*

ENTRY
The *ENTRY window* begins right after the arrow near the bottom of your display screen and continues to the vertical bar on the last line of the screen. This area is where you normally type in TSO/E commands.

PASSWORD
The window after the vertical bar is the *PASSWORD window* and can be used to type in passwords for your data sets. The information you type in the PASSWORD window will not appear on the display screen or in your session journal.

Session Manager also uses the PASSWORD window to display error messages to you.



Program Function (PF) Keys

Some of the defaults for PF keys in Session Manager are different from defaults used in other products. The following summarizes the PF key default values in Session Manager:

PF Key
Function

PF2 (or 14)
This key sets the value for the *scroll amount window* located on the lower right-hand side of your display screen (SCROLL===>). Initially, this value is set to HALF. This means that if you press one of the scroll PF keys, the MAIN window scrolls your session journal one-half page.

You can change the scroll amount by typing a new value and pressing PF2 (or 14).

The valid scroll amounts are:

- Page: Scrolls one page, with page being the number of lines in your MAIN window. You can abbreviate page by just typing in P.

- Half: Scrolls one-half page. You can abbreviate half by just typing in H.
- Max: Scrolls the maximum length or width of your session journal depending on which scroll PF key you press. You can abbreviate max by just typing in M.
- A number from 1 to 999999: Scrolls that number of lines forward or backward when you press PF7 (or 19) or PF8 (or 20).
- A number from 1 to 32767: Scrolls that number of columns left or right when you press PF10 (or 22) or PF11 (or 23).

If you enter a not valid value, or press PF2 (or 14) without typing in a value, Session Manager places the value you typed (or blanks if you did not type anything) in the scroll amount window as if it were valid. You will not get an error message until you attempt to use one of the scroll PF keys. You can easily change the scroll amount by typing in a valid value and pressing PF2 (or 14).

PF5 (or 17)
Occasionally, you might want to locate a particular line of output in your session journal. Session Manager defines PF5 (or 17) to do this search for you. All you have to do is type in a character string from the line you want to find and press PF5 (or 17). You can also repeat the search by pressing the key again. Session Manager remembers what you typed.

Some important facts to remember when using this key are:

- You must type the character string exactly as it appears in your session journal. For example, if the character string you want to find is in uppercase letters, you must type it in uppercase letters.
- When you use PF5 (or 17) to find a line in your session journal, Session Manager begins the search from where your MAIN window is located and goes *backward* towards the top. Therefore, it might be best to position the MAIN window at the bottom of your session journal so that occurrences of the string are not overlooked.

PF6 (or 18)
Session Manager defines PF6 (or 18) to change the lines that the CURRENT window displays. When you first log on, this window lets you see the last two lines in your session journal. If you press PF6 (or 18), the CURRENT window changes to display the last two lines of input. This can be useful if you want to remember the last two commands or lines of input you entered without having to scroll the MAIN window back over your session journal. So, in effect, PF6 (or 18) "flip-flops" the CURRENT window to let you see either the last two lines that you typed in or the last two lines of output in your session journal.

The remaining PF keys are the *scroll PF keys*.

The following four scroll PF keys move the MAIN window a specific amount over the information in your session journal. How far the window moves depends on the value that is set in the scroll amount window.

PF Key
Function

PF7 (or 19)
Causes the MAIN window to scroll **backward** towards the top of your session journal.

PF8 (or 20)
Causes the MAIN window to scroll *forward* towards the bottom of your session journal.

PF10 (or 22)
Causes the MAIN window to scroll to the *left* side of the information in your session journal.

PF11 (or 23)
Causes the MAIN window to scroll to the *right* side of the information in your session journal.

If you have lines of output that are wider than your display screen, use this key to look at the rest of the output.

If, after doing several things at your terminal, you want to look at the information at the very beginning or the very end of your session journal, you do not have to repeatedly press PF7 (or 19) or PF8 (or 20) to scroll to this information. Session Manager defines the following two scroll PF keys to move the MAIN window either to the top or the bottom of your session journal.

PF Key
Function

PF9 (or 21)
Causes the MAIN window to scroll **backward** all the way to the top of your session journal.

PF12 (or 24)
Causes the MAIN window to scroll **forward** all the way to the bottom of your session journal.

Locking and Unlocking the MAIN Window

When your MAIN window is full of information, you generally press the Enter key to see another screen of information. The Enter key moves the session journal forward towards the most current information.

If you use the FIND PF key (PF5 or PF17) or one of the scroll PF keys (PF7 or PF19 through PF11 or PF23) to go backward in the session journal, the Enter key no longer moves the session journal forward. When this happens, the MAIN window is considered **locked** and the word "locked" appears in the MAIN field in the lower right-hand corner of the display screen. You can, however, continue to use the FIND PF key (PF5 or PF17) and the scroll PF keys (PF7 or PF19 through PF11 or PF23) to move forward and backward in your session journal, and you can continue to enter commands in the CURRENT window. The CURRENT window never locks.

To **unlock** the MAIN window, press PF12 (or 24) to position you at the most current part of your session journal. The word "unlocked" then appears in the MAIN field in the lower right-hand corner of the display screen and the Enter key again moves the session journal forward.

To summarize:

- The MAIN window **locks** when you press PF5 (or 17) or any PF key from PF7 (or 19) to PF11 (or 23).
- To **unlock** the MAIN window, press PF12 (or 24).
- The CURRENT window never locks.
- The status of whether or not the MAIN window is locked is displayed in the MAIN field of the STATUS window.

Using Displayed Information to Form New Input

You can enter TSO/E commands or lines of input anywhere on the display screen except on the numbered line or in the STATUS windows. Anything you type and enter in the MAIN, CURRENT, PASSWORD, and ENTRY windows is sent to TSO/E to be processed.

When you enter more than one command or line of input as a group, TSO/E processes these lines from the top of the display screen to the bottom. Therefore, the order in which you type the lines is not necessarily the order in which they will be processed. For example, if you first type a command in the ENTRY window, and then move the cursor up into the MAIN window and type another command, TSO/E executes the command in the MAIN window before it executes the one in the ENTRY window.

You can also save time and keystrokes by using currently displayed lines to create new commands and input. Here are two ways you can do this:

1. To reissue a command, retype a blank on the same line that the command is on or type over one of the characters in the command and press the Enter key. Session Manager sends this line to TSO/E just as if you had typed in the entire line.
2. If you are looking at a list of data sets (for example, output from the LISTCAT command, which causes a list of your data sets to be displayed), and you want to issue a command using one or more of these data sets, simply move the cursor to the appropriate line on the display screen and insert your command around the data set name(s). You do not have to retype the data set name(s).

Effects of Entering a Null Line

When you are communicating with TSO/E and Session Manager, you may want to send a null line to TSO/E. To enter a null line, position the cursor at the beginning of a field on the screen, press ERASE EOF, and then Enter.

If you press the Enter key without pressing ERASE EOF or without modifying a field, a null line will not be sent to TSO/E. Instead, Session Manager interprets this action as a request to automatically scroll unlocked windows over lines of output that were generated by your TSO/E application since the keyboard was last unlocked.

Sending a changed field to TSO/E and sending a null line can produce the same result from Session Manager. For example, a full-screen application, such as ISPF, may suspend its operation to permit line-mode output to be written to Session Manager data streams. You may then see this message:

```
ADF041A ENTER A NULL LINE TO RETURN TO FULL-SCREEN PROGRAM
```

Message ADF041A may be displayed before you have had an adequate opportunity to read the line-mode output. You can scroll through the line-mode output by sending scroll commands to Session Manager or by using the program function keys that allow you to scroll.

When you want to reactivate the full-screen program, you can enter a null line. However, if you change a field intended for TSO/E (for example, if you enter a TSO/E command) and press the Enter key, the data is not sent to TSO/E, but rather it is interpreted by Session Manager as a request to reactivate the full-screen program. In this case, Session Manager interprets the changed field as a null line.

Getting a Copy of Your Session Journal

You can print a copy of the information in your session journal by typing

```
SMCOPY
```

or

```
SMC
```

and pressing the Enter key.

This TSO/E Session Manager command sends a copy of all the information in your session journal to the system printer.

A complete description of the SMCOPY command can be found in *z/OS TSO/E Command Reference*.

A session journal is large enough to be sufficient for most TSO/E users. However, it is limited in size. If you tend to do work that generates a lot of output to your session journal (for example, listing large data sets or directing compiler output to your terminal), you can use up your entire session journal. When your session journal fills up, new output begins to overlap the lines at the top. This is called "wrapping" and you will probably not know it has happened until you scroll backward and can't find a line at the beginning of your session journal. If you think this might happen to you and you want to remember what you did in the beginning of your terminal session, periodically type in the SMCOPY command to make a copy of the information before it wraps.

Entering Session Manager Commands

You can enter Session Manager commands by:

- Pressing the CLEAR key and entering a command anywhere on the screen
- Pressing a program function (PF) key set up to issue a command
- Executing a TSO/E CLIST, which contains commands
- Defining the command as the text-string of the TSO/E SMPUT command.

Regardless of how you enter the commands, the following rules apply:

- You can enter multiple Session Manager commands on one line provided you separate them with a semicolon (;). The number of characters on any one line cannot exceed 512. When multiple commands are entered on a line, an error in one command does not prevent the remaining commands from executing.
- For a Session Manager command to execute, it must be placed in the Session Manager input stream.
- Any change you make to the definitions of the windows, cursor, PF keys, session functions, streams, or the terminal, remain in effect for your terminal session. You can place these definitions in a CLIST to be executed each time you log on.

Controlling The Session Manager Environment

One of the features of Session Manager is that you can change the default environment (the program function (PF) keys, streams, and screen layout) to suit your specific needs. This part explains how Session Manager commands are used to set up this environment, how you can use the commands to change the environment, and provides examples of when these changes might be appropriate. For complete descriptions of all commands referred to in this part, see *z/OS TSO/E Command Reference*.

Streams

Session Manager keeps several records of the different things that happen during your terminal session - all the commands, instructions, and input you enter and all the output from commands and messages that the system issues. These records are **streams**.

A stream is similar to a long sheet of paper that fills up with information as your terminal session progresses. Session Manager places each line of information entering a stream in the next available line starting at the top.

Types of Streams

The three types of Session Manager streams are **input**, **output**, and **extra**.

Input Stream

An input stream contains information that you want interpreted as commands or input to programs.

Output Stream

An output stream contains the output from commands and any messages from other TSO/E users, the operator, or from background jobs. You can also copy information from an input stream into an output stream, thereby creating a complete record of all commands entered and all of the output from those commands.

Extra Stream

You can copy information, such as the output from certain commands, into an extra stream. In the default environment, an extra stream is used to contain the lines, windows, and arrows that visually divide the display screen. You cannot specify an extra stream on any of the Session Manager commands when an input or output stream is required.

Changing the Streams

In the IBM-supplied default environment, nine streams have been defined. You cannot change the names of any of the streams, but you can use the CHANGE.STREAM command to:

- Erase the information in a stream
- Change whether the terminal's audible alarm sounds when information enters the stream.

The following figure summarizes the attributes of all of the streams in the default environment. It identifies the stream name, type, size (in lines and bytes) and indicates whether the audible alarm sounds when information enters the stream.

Table 3. Streams

Stream Name	Type	Linesize	Bytesize	Alarm
TSOIN	Input	305	8192	No
TSOOUT	Output	4005	147456	No
SMIN	Input	305	8192	No
SMOUT	Output	155	4096	No
MESSAGE	Output	55	1024	Yes
EXTRA1	Output	405	32768	No
EXTRA2	Extra	105	1024	No
EXTRA3	Extra	105	1024	No
HEADER	Extra	55	1024	No

Session Functions

While working at the terminal, you are communicating with TSO/E and Session Manager. In addition, you can receive messages from other TSO/E users, the operator, and from jobs you're executing. These three means of communication (TSO/E, Session Manager, and messages) are called *session functions*. The session functions receive the input you enter at the keyboard and generate output to you. Session Manager places the input and output in streams, either the default streams or the streams that you specify on the CHANGE.FUNCTION command.

TSO/E has an input and an output stream:

- TSOIN is the input stream for the TSO/E function. Information placed in this stream is interpreted as TSO/E commands or input to programs. While a command is executing in this stream, Session Manager highlights it. Otherwise, when the command has completed execution, it is displayed at normal intensity.
- TSOOUT is the output stream for the TSO/E function. All output generated by the TSO/E commands and programs is placed in this stream. The TSOOUT stream also contains a copy of the TSO/E commands from the TSOIN stream, thus creating a complete record of your TSO/E session. (The commands from the TSOIN stream appear highlighted in this stream.) In addition, the MESSAGE (MSG) function places in the TSOOUT stream any messages you receive from other TSO/E users, the operator, or from background jobs. As a result, the messages are interleaved with the TSO/E session input and output.

Session Manager (SM) Function

Session Manager also has an input and output stream:

- SMIN is the input stream for the SM function. Information placed in this stream is interpreted as Session Manager commands.
- SMOUT is the output stream for the SM function. Any error messages from the Session Manager commands are placed in this stream. The SMOUT stream also contains a copy of the Session Manager commands from the SMIN stream. However, you cannot see the commands at your terminal because they are copied at an intensity of 0. This is done so that the PASSWD window on your screen only displays the error messages.

Message (MSG) Function

You must place the messages you receive from other TSO/E users, the operator, or from jobs you're executing in an output stream. In the default environment, the TSOOUT stream contains any messages you receive interleaved with your TSO/E input and output. The message (MSG) function cannot have an input or a copy stream.

Changing the Session Function Streams

The following figure summarizes the session functions and their attributes. You cannot specify an input stream for the message (MSG) function. Therefore, this area has been left blank in the figure below.

Intensity (INT) refers to the brightness at which the information is displayed in a stream.

0
The information in the stream is not displayed. You can see the line that the information occupies, but the information itself is invisible.

1
The information in the stream is displayed at normal intensity.

2
The information in the stream is highlighted.

ALARM indicates whether the terminal's audible alarm sounds when information enters the stream.

Session Function	**Input Stream**	**Output Stream**					
	Name	**Alarm**	**Where Copied**	**INT**	**Name**	**INT**	**Alarm**
TSO/E	TSOIN	NO	TSOOUT	2	TSOOUT	1	NO
SM	SMIN	NO	SMOUT	0	SMOUT	2	YES
MSG	-	-	-	-	TSOOUT	2	YES

You can change the input or output streams for any of the session functions, any attributes relating to those streams, and whether the alarm is to sound. For example, you might want your messages to go to the MESSAGE stream instead of the TSOOUT stream or you might not want the TSOIN stream copied to the TSOOUT stream. With the CHANGE.FUNCTION command, you can specify:

- The output stream for a session function and the intensity at which the information is to be displayed
- The input stream for a session function
- Whether information from an input stream is to be copied to an output stream and the intensity at which the information is to be displayed.
- Whether the audible alarm is to sound when information enters an input or output stream.

The commands that define the streams and their characteristics for each of the session functions in the default environment are:

```
CHANGE.FUNCTION TSO INPUT(TSOIN) COPY(TSOOUT 2)
   OUTPUT(TSOOUT 1) ALARM(NO)
CHANGE.FUNCTION MSG OUTPUT(TSOOUT 2) ALARM(OUTPUT)
CHANGE.FUNCTION SM INPUT(SMIN) COPY(SMOUT 0)
 OUTPUT(SMOUT 2) ALARM(OUTPUT)
```

For more information, see the syntax description of the CHANGE.FUNCTION command in *z/OS TSO/E Command Reference*.

Changing the Screen Layout

The default screen layout consists of the definitions for the windows and the location of the cursor on the display screen. This part describes:

- The definition and characteristics of a window
- How to change and delete windows on the screen
- How to scroll a window
- How to lock and unlock a window
- How to display streams
- How to change the location of the cursor on the screen.

Defining a New Window

A **window** is a physical area of the display screen that you can use to type into and/or display a particular stream.

You can define a window by giving it all of the desired attributes at once using the DEFINE.WINDOW command, or you can issue that command followed by one or more CHANGE.WINDOW commands to build the attributes for the window. The attributes for the window remain in effect until modified via the CHANGE.WINDOW command.

Following are the attributes you need to know to define a window. The operand for specifying each attribute is enclosed in parentheses.

- The name of the window (window-name).
- The location of the window on the display screen (row, column, lines, and width).

  "Row" refers to the number of the top line in the window. "Column" specifies which column of the screen the left side of the window is to occupy. "Lines" specify the number of lines in the window. "Width" specifies the number of character positions in each line. If you specify a negative value for either "row" or "column", the Session Manager interprets the value as relative to the bottom or right side of the screen, respectively.

  You can use the word MAX for either the "line" or "width" operand instead of specifying a number. If you specify MAX for the line operand, the window contains the remaining lines on the screen or until a line is encountered that has been defined for another window. If you specify MAX for the width operand, the window's width is determined by the number of character positions in the first line of the window.

  In the example below, the TEST window was defined first beginning in row 22, column 1 and the SAMPLE window begins in row 1, column -20. If you define a new window beginning in row 1, width MAX, and lines MAX, the window covers rows 1 through 21 and columns 1-59.

  Note: The screen in the example has a column width of 80.

  You can also specify WRAP for the width value instead of MAX or a number. In this case, the width of the window starts from the column value specified on the command and continues to either the beginning of the next window or to the last row and column on the screen. The WRAP operand can only be used when the value for "lines" is 1.

  Note: The first character position in a window is used as a terminal attribute byte and is protected. Therefore, a window defined with a width of 1 is useless.
- Whether you want the terminal's audible alarm to sound when the window scrolls to display new information in the stream (ALARM).
- How long you want a window to be held in place before Session Manager scrolls it (HOLD). The window can remain at one position a specified number of seconds or until you press the Enter or any program function (PF) key. While the window is held in place, the keyboard is locked.
- How many lines are to be repeated in the window when it scrolls over the stream (OVERLAP). For example, if you specify OVERLAP(2) in defining a window, the bottom or top 2 lines in the window are repeated when you move the window forward or backward, respectively.

  If you move the above window forward a complete page, lines 4 and 5 are repeated.
- Whether you can enter data in the window (PROTECT).

- The name of the stream that is to receive the information typed in the window and the intensity at which the information is to be displayed (TARGET).
- How much information is to enter the specified stream before the window scrolls to display it (UPDATE). When a window is displaying the bottom of a stream, it waits for new information to enter the stream before it moves. With the UPDATE operand, you can specify whether the window is to display every line of information as it enters the stream, every page of information, or just the newest information. In the latter case, some lines of information might be skipped over because the window scrolls directly to the bottom of the stream.
- The name of the stream the window is to display (VIEW).

For more information on the above operands, see the syntax description for the DEFINE.WINDOW command in *z/OS TSO/E Command Reference*.

When you define a window on the display screen, the physical location of the window cannot exceed the physical limits of the display screen unless you define the window as one line that wraps the screen. In addition, a new window cannot overlap the physical location of an existing window.

Changing a Window

Instead of defining a new window on the display screen, you might want to simply change the attributes of an existing window. The following attributes of a window can be changed with the CHANGE.WINDOW command:

- Whether you want the terminal's audible alarm to sound when the window scrolls to display new information in the stream
- How long the window is to be held in place before the Session Manager scrolls it
- How many lines are to be repeated in the window when it scrolls over the stream
- Whether you can enter data in the window
- The name of the stream that is to receive the information typed in the window and the intensity at which the information is to be displayed
- How much information is to enter the specified stream before the window scrolls to display it
- The name of the stream the window is to display.

You cannot change the window name, row, column, lines, or width values for an existing window unless you delete the entire window and redefine it.

Note: “Defining a New Window” on page 171 describes each of these attributes.

Deleting a Window

You can use the DELETE.WINDOW command to delete one or all of the windows on the display screen. The space previously occupied by the window(s) can then be defined as another window or windows.

If you delete all of the windows, you cannot enter input on the screen unless you press the CLEAR key. To reestablish the default screen layout after deleting any of the windows, issue the RESET command.

Displaying Streams

Each window in the default environment displays a particular stream. The name of the stream is set in the DEFINE.WINDOW command with operand VIEW(stream-name). You can change the stream by using the same operand on the CHANGE.WINDOW command. The following paragraph describes some instances in which you might want to change the stream that the MAIN window displays.

In the default environment, the MAIN window displays the TSOOUT stream that contains a copy of all TSO/E commands entered, the output from those commands, and any messages received from other TSO/E users, the operator, or background jobs. Occasionally, you might want to have the MAIN window display only the TSO/E commands entered up to a particular point in your terminal session. In this case, you can change the MAIN window to display the TSOIN stream and use the scroll PF keys to review the entire stream.

Having the MAIN window display the TSOIN stream is also useful when executing and monitoring many commands at one time. For example, if you wanted to compile several programs, you can enter all of the commands to do the compilations at one time and then monitor the execution of those commands (Session Manager highlights each command as it is being executed in the TSOIN stream).

To display a list of the windows defined in the default environment and their attributes, use the QUERY.WINDOW or QUERY.TERMINAL command.

Scrolling a Window

Several Session Manager commands allow you to move a window over the stream it is displaying. This process of moving a window over a stream is called *scrolling* and is controlled with the FIND and SCROLL commands.

With the FIND command, Session Manager scrolls the window so that the specified text-string is the top line in the window. With the SCROLL commands, you can control the direction and amount Session Manager scrolls the window. Both of these commands automatically lock the window in place after it has moved. The window remains at the locked position until you issue another SCROLL or FIND command or an UNLOCK command.

In the default environment, the program function (PF) keys are defined to issue the SCROLL and FIND commands. For a description of how these keys are defined, see “Changing Program Function (PF) Key Definitions” on page 176.

How to Lock or Unlock a Window

Whenever you or the system enters data in a stream, you can easily scroll towards the bottom of the stream by pressing the Enter key each time the window fills up with information. Sometimes, however, you don't want your data to move, for example, when you are scrolling or using the FIND command to locate a specific text-string. Therefore, Session Manager automatically locks the window when you issue the FIND or SCROLL commands.

When a window is locked, it means that the window is frozen at its current position and will not move to display new information even if you press the Enter key. In the default environment where several of the program function (PF) keys issue the FIND and SCROLL commands, pressing the key causes the default window (MAIN) to lock. This window locks so that when you use a PF key to scroll over the information in a stream, the window remains at that position until you are ready to move it. A locked window, however, does not mean that new information cannot enter the bottom of the stream.

An unlocked window in the default environment means that each time you press the Enter key, the MAIN window automatically moves to the next half-page of information.

You can unlock a window at its present position, at the bottom of the stream, or at the information it was displaying when it was last unlocked by using the UNLOCK command. After issuing this command, you can automatically move the window forward over the stream by pressing the Enter key.

The Session Manager Display Screen

Figure 12 on page 174 illustrates the windows on the Session Manager default display screen. The following figure lists the attributes of each window (the operands used to define it). The windows were defined in the order listed at the top of the figure to take advantage of the MAX and WRAP operands on the DEFINE.WINDOW command.

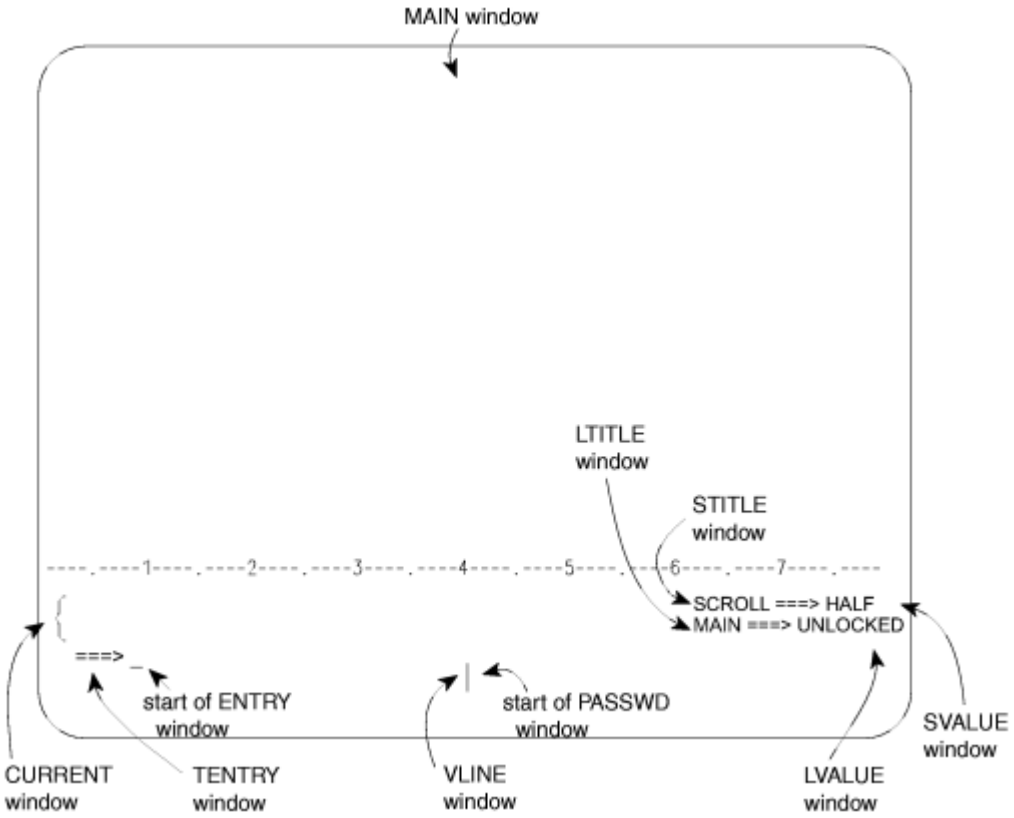


Figure 12. Session Manager Display Screen

Table 4. Session Manager Default Display Screen Window Definitions

OPER-ANDS	WINDOW NAME										
	LINE	STITLE	SVALUE	LTITLE	LVALUE	VLINE	PASSWD	CUR- RENT	TENTRY	ENTRY	MAIN
row	-5	-4	-4	-3	-3	-1	-1	-4	-2	-2	1
column	1	-18	-6	-18	-9	-40	-38	1	1	6	1
lines	1	1	1	1	1	1	1	2	1	1	MAX
width	MAX	12	6	9	9	2	38	MAX	5	WRAP	MAX
ALARM	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
HOLD	0	0	0	0	0	0	0	0	0	0	INPUT
OVER- LAP	0	0	0	0	0	0	0	0	0	0	0
PRO- TECT	YES	YES	YES	YES	YES	YES	NO	NO	YES	NO	NO
TARGET STREAM			EXTRA3				TSOIN	TSOIN		TSOIN	TSOIN
UPDATE	NEWEST	NEWEST	NEWEST	NEWEST	NEWEST	NEWEST	NEWEST	NEWEST	NEWEST	NEWEST	LINE
VIEW STREAM	HEADER line 2	HEADER line 6	EXTRA3	HEADER line 9	HEADER line 7 line 8	HEADER line 4	SMOUT	TSOOUT line 3	HEADER line 1	HEADER line 10	TSOOUT

Window
Description

MAIN
The MAIN window is the large portion of the display screen above the numbered line. This window displays the TSOOUT stream which contains a copy of all the TSO/E commands you enter, all output from those commands, and any messages you receive from other TSO/E users, the operator, or background jobs. If you enter information in the MAIN window, Session Manager sends it to the TSOIN stream to be executed as a TSO/E command.

LINE
This window displays the numbered line located on line 2 in the HEADER stream. If you try to type in the LINE window, the keyboard locks. Press the RESET key to unlock the keyboard.

CURRENT
Just below the numbered line is the CURRENT window. Initially, this window displays the last two lines in the TSOOUT stream. If you press PF6/18, the CURRENT window changes to display the last two lines in the TSOIN stream (the last two commands you entered). For a complete description of how PF6/18 changes the CURRENT window between displaying the TSOOUT and TSOIN streams, see “Changing Program Function (PF) Key Definitions” on page 176.

If you enter information from the CURRENT window, Session Manager sends it to the TSOIN stream to be executed as a TSO/E command.

TENTRY
The TENTRY window displays an arrow located on line 1 in the HEADER stream. If you try to type in this window, the keyboard locks. Press the RESET key to unlock the keyboard.

ENTRY
The ENTRY window begins after the TENTRY window and continues to the vertical bar on the last line of the screen. This window "wraps the screen" which means that it is defined as one line that physically covers two or more lines on the screen. The ENTRY window is where you normally enter TSO commands to go to the TSOIN stream although you can enter them from the MAIN, CURRENT, and PASSWD windows. This window displays a blank line located on line 10 in the HEADER stream.

STITLE
The STITLE window displays the word "SCROLL" located on line 6 in the HEADER stream. This window, in combination with the SVALUE window, lets you know what the scroll value is for the scroll PF keys. If you try to type in the STITLE window, your keyboard locks. Press the RESET key to unlock the keyboard.

SVALUE
This window displays the last line in the EXTRA3 stream which contains the scroll amount value for the scroll PF keys. Initially, this value is set at HALF, which means that if you press one of the scroll PF keys, the MAIN window scrolls one half-page. You can change the scroll amount value by typing a new value in the MAIN, CURRENT, ENTRY, or PASSWD windows and pressing PF2/14. If you try to type directly in the SVALUE window, the keyboard locks. Press the RESET key to unlock the keyboard.

For the valid scroll amounts, see “Program Function (PF) Keys” on page 164. For a complete description of how PF2/14 sets the scroll amount value for the scroll PF keys, see “Changing Program Function (PF) Key Definitions” on page 176.

LTITLE
The LTITLE window displays the word "MAIN" on line 9 of the HEADER stream. This window, in combination with the LVALUE window, lets you know if the MAIN window is locked or unlocked. If you try to type in the LTITLE window, the keyboard locks. Press the RESET key to unlock the keyboard.

LVALUE
The LVALUE window displays either "UNLOCKED" (on line 7) or "LOCKED" (on line 8) in the HEADER stream. These words refer to the status of the MAIN window. When you issue a SCROLL or FIND command, Session Manager automatically locks the MAIN window. The PF keys that issue these commands also issue a command to move the LVALUE window over the word "LOCKED". To unlock the MAIN window, issue an UNLOCK command or press PF12/24, which issues that command followed by a command that moves the window over the word "UNLOCKED". For more information, see “Changing Program Function (PF) Key Definitions” on page 176 and “How to Lock or Unlock a Window” on page 173.

VLINE
The VLINE window displays a vertical bar on line 4 in the HEADER stream. If you try to type in this window, your keyboard locks. Press the RESET key to unlock the keyboard.

PASSWD
Located after the VLINE window is the PASSWD window. This window can be used to type in any information you do not want displayed on the screen (such as the passwords for data sets). Any

information you enter from this window goes to the TSOIN stream. The PASSWD window displays the SMOUT stream. If you receive an error message from a Session Manager command, you will see the message in this window.

Changing the Location of the Cursor

The CHANGE.CURSOR command lets you change the location to which the cursor returns after you press a program function (PF) key, the Enter key, the CLEAR key, the attention (PA1) key, or the cancel (PA2) key. You can establish a **permanent** or **temporary** location.

If you define a permanent location, the cursor returns to that location each time you press one of the mentioned keys. If you define a temporary location, the cursor moves to and remains at that location until the next keyboard entry. After the keyboard entry, the cursor moves to the permanent location.

The cursor is defined as being on a particular row and column within a window. In the default screen layout, the permanent location of the cursor is defined to be row 1 column 1 of the ENTRY window. A temporary cursor location is not defined.

Changing the Mode

If you want to run VS/APL using the APL character set under Session Manager, you must issue the Session Manager command CHANGE.MODE APL. See *z/OS TSO/E Command Reference* for more information about the CHANGE.MODE command.

Changing Program Function (PF) Key Definitions

Program function (PF) keys can be defined as TSO/E or Session Manager commands, input to programs, or text-strings that comment your streams. Use the CHANGE.PFK command to modify the meaning of any of the PF keys. This command requires that you enter the number of the PF key to be changed, the definition-text-string for the key, and the name of the stream where the definition-text-string is to be placed when the PF key is pressed.

You can make a PF key a TSO/E command (or commands) by defining the definition-text-string as a valid TSO/E command or subcommand and specifying TSOIN as the target stream-name. This is useful when working with programs that prompt for input or commands that have subcommands. For example, you can define a PF key as often-entered subcommands of the TSO/E TEST command, such as SAVE or GO.

You can make a PF key a Session Manager command (or commands) by defining the definition-text-string as a valid Session Manager command and specifying SMIN as the target stream-name. This is done in the default environment to allow you to use the PF keys for scrolling.

If you have the definition-text-string sent to another stream (such as the TSOOUT stream), the PF keys can be used to comment your streams.

When you press a PF key, Session Manager:

- Places each line of information typed on the screen since the last time you pressed the Enter, CLEAR, PA1, PA2, or any PF key in the appropriate stream, just as if you had pressed the Enter key.
- Places the definition-text-string of the PF key definition in the target stream.

You can also define PF keys to perform symbolic substitution when they are pressed. When you define the PF key, you can specify **symbolic arguments** within the definition-text-string and specify the SUBSTITUTE keyword operand to indicate that symbolic substitution is to be performed. Each symbolic argument is denoted by an ampersand (&) followed by an integer (or an asterisk). For example,

```
&1., &2., &3., … &n., and &*.
```

If the ampersand appears elsewhere in the definition-text-string when SUBSTITUTE is specified, it must be doubled.

You must provide parameters (tokens) to this symbolic substitution process by entering them on a line of the screen (separated by one or more blanks), and pressing the PF key.

Session Manager performs the symbolic substitution for each new line of information found on the screen when you press a PF key. It:

- Scans the line of information once from left to right and breaks the line into as many words (tokens) as there are numbered symbolic arguments in the PF key definition-text-string. This process is called tokenization. The remainder of the text in the line becomes the "*" token. If there are no numeric symbolic arguments in the text, the entire line of information becomes the "*" token.
- Substitutes the words (tokens) into the PF key definition-text-string, replacing the symbolic arguments. If there are more symbolic arguments than words, null characters are substituted for those arguments. If there are more words than symbolic arguments, the extra words are ignored. If there are no modified fields on the screen, null characters are substituted for each argument in the text-string.
- Places the resulting definition-text-string in the next available line of the target stream.

The periods immediately following the numeric symbolic arguments are used to distinguish the arguments from the characters immediately following them in the definition-text-string. The periods are not needed if the character after the argument is not a digit.

The example section of the CHANGE.PFK command includes several examples of how the PF keys can be defined.

Most PF keys in the default environment are defined as a string of Session Manager commands that are placed in the SMIN stream.

The following table summarizes the default definitions of the PF keys. PF keys 13-24 have the same definition as PF keys 1-12.

Table 5. Program Function (PF) Key Definitions

PF Key/PF Key Default Definition	PF Key/PF Key Default Definition	PF Key/PF Key Default Definition
PF1/13 Not Defined	PF2/14 Set Scroll Amount	PF3/15 Not Defined
PF4/16 Not Defined	PF5/17 Find/Repeat Find	PF6/18 Change CURRENT Window
PF7/19 Scroll Backward	PF8/20 Scroll Forward	PF9/21 Scroll Top
PF10/22 Scroll Left	PF11/23 Scroll Right	PF12/24 Scroll Bottom Unlock

The following section describes how the PF keys are defined in the default environment. Most of the keys are defined as several Session Manager commands that work together to create the desired effect. Unless you are using the split screen capability or have changed the definition for the default window, all references to the default window mean the MAIN window. For a complete description of each command, refer to *z/OS TSO/E Command Reference*.

PF1/13

PF1/13 is not defined in the default environment. If you press this key, Session Manager does the following:

- Places the message "PF1 NOT DEFINED" or "PF13 NOT DEFINED" in the SMOUT stream
- Moves the PASSWD window so that the window displays the message.

The commands that issue the message and move the PASSWD window over the SMOUT stream are:

```
CHANGE.PFK 1 'PFK 1 NOT DEFINED' SMOUT;
             CHANGE.WINDOW PASSWD
             VIEW(SMOUT)
```

PF2/14

PF2/14 accepts the value typed on the screen for the scroll amount field and places this value highlighted in the EXTRA3 stream. It then changes the definitions of the scroll PF keys (PF7/19, PF8/20, PF10/22, and PF11/23) to do the following:

- Set the scroll value to zero. This is done so that the default window will not move if a not valid value is entered.
- Scroll the amount entered. Session Manager substitutes the value entered in the place of the symbolic argument in the AMOUNT operand.
- Scroll the LVALUE window over line 8 in the HEADER stream so the word "LOCKED" is displayed.
- Unlock the terminal keyboard so additional scrolling PF keys can be pressed.
- Scroll the LINE window the entered amount (for PF keys 10/22 and 11/23 only).

The commands that define PF2/14 are:

```
CHANGE.PFK 2 'PUT ''&1'' EXTRA3 INTENSITY(2);
              CHANGE.PFK 7 ''SCROLL.BACK 0;
               SCROLL.BACK AMOUNT(&1);
               SCROLL.ABSOLUTE 8 LVALUE;
               CHANGE.TERMINAL CONTROL(0)''
               SMIN;
              CHANGE.PFK 8 ''SCROLL.FORWARD 0;
               SCROLL.FORWARD AMOUNT(&1);
               SCROLL.ABSOLUTE 8 LVALUE;
               CHANGE.TERMINAL CONTROL(0)''
               SMIN;
              CHANGE.PFK 10 ''SCROLL.LEFT 0;
               SCROLL.LEFT AMOUNT(&1);
               SCROLL.LEFT LINE AMOUNT(&1);
               SCROLL.ABSOLUTE 8 LVALUE;
               CHANGE.TERMINAL CONTROL(0)''
               SMIN;
              CHANGE.PFK 11 ''SCROLL.RIGHT 0;
               SCROLL.RIGHT AMOUNT(&1);
               SCROLL.RIGHT LINE AMOUNT(&1);
               SCROLL.ABSOLUTE 8 LVALUE;
               CHANGE.TERMINAL CONTROL(0)''
               SMIN' SMIN SUBSTITUTE
```

PF3/15

PF3/15 is not defined in the default environment. If you press this key, Session Manager does the following:

- Places the message "PF3 NOT DEFINED" or "PF15 NOT DEFINED" in the SMOUT stream
- Moves the PASSWD window so that it displays the message

The commands that issue the message and move the PASSWD window over the SMOUT stream are:

```
CHANGE.PFK 3 'PFK 3 NOT DEFINED' SMOUT;
              CHANGE.WINDOW PASSWD VIEW(SMOUT)
```

PF4/16

PF4/16 is not defined in the default environment. If you press this key, Session Manager does the following:

- Places the message "PF4 NOT DEFINED" or "PF16 NOT DEFINED" in the SMOUT stream
- Moves the PASSWD window so that it displays the message

The commands that issue the message and move the PASSWD window over the SMOUT stream are:

```
CHANGE.PFK 4 'PFK 4 NOT DEFINED' SMOUT;
              CHANGE.WINDOW PASSWD VIEW(SMOUT)
```

PF5/17

PF5/17 is the find/repeat find key. It does the following:

- Locks the default window by issuing the SCROLL command specifying a value of 0.

- Issues the FIND.BACKWARD command substituting the information typed on the screen as the character string to be found. If you press PF5/17 without typing anything on the screen, the previous find command is repeated.
- Sets the terminal keyboard control to unlock after 5 seconds.
- Scrolls the LVALUE window to display the word "LOCKED" on line 8 in the HEADER stream.

The commands that define PF5/17 are:

```
CHANGE.PFK 5 'SCROLL.BACKWARD 0;
              FIND.BACKWARD ''&*'';
              CHANGE    TERMINAL CONTROL(5);
              SCROLL.ABSOLUTE 8 LVALUE' SMIN
              SUBSTITUTE(&)
```

PF6/18

PF6/18 changes the CURRENT window to display either the last two lines in the TSOOUT stream or the last two lines in the TSOIN stream. Initially, the window displays the last two lines in the TSOOUT stream. When you press PF6/18, it does the following:

- Saves all of the current PF key definitions
- Changes the CURRENT window to display the TSOIN stream
- Redefines itself so that the next time you press the key, the CURRENT window again displays the TSOOUT stream and the PF key definitions are restored

The commands that define PF6/18 are:

```
CHANGE.PFK 6 'SAVE.PFK;
              CHANGE.WINDOW CURRENT
              TARGET(TSOIN) VIEW(TSOIN);
              CHANGE.PFK 6 ''CHANGE.WINDOW
              CURRENT TARGET(TSOIN)
              VIEW(TSOOUT);
              RESTORE.PFK '' SMIN ' SMIN
```

PF7/19

PF7/19 does the following:

- Scrolls the default window backward a half-page over the stream it is displaying (As part of the SCROLL command, Session Manager locks the window.)
- Unlocks the keyboard
- Scrolls the LVALUE window to display the word "LOCKED" on line 8 in the HEADER stream

The commands that define PF7/19 are:

```
CHANGE.PFK 7 'SCROLL.BACKWARD AMOUNT(HALF);
               CHANGE.TERMINAL CNTL(0);
               SCROLL.ABSOLUTE 8 LVALUE' SMIN
```

PF8/20

PF8/20 does the following:

- Scrolls the default window forward a half-page over the stream it is displaying (As part of the SCROLL command, Session Manager locks the window.)
- Unlocks the keyboard
- Scrolls the LVALUE window to display the word "LOCKED" on line 8 in the HEADER stream

The commands that define PF8/20 are:

```
CHANGE.PFK 8 'SCROLL.FORWARD AMOUNT(HALF);
               CHANGE.TERMINAL CNTL(0);
               SCROLL.ABSOLUTE 8 LVALUE' SMIN
```

PF9/21

PF9/21 does the following:

- Scrolls the default window to the oldest information in the stream it is displaying (As part of the SCROLL command, Session Manager locks the window.)
- Unlocks the terminal keyboard so that additional PF keys can be used
- Scrolls the LVALUE window to display the word "LOCKED" on line 8 in the HEADER stream

The commands that define PF9/21 are:

```
CHANGE.PFK 9 'SCROLL.OLDEST;
              CHANGE.TERMINAL CNTL(0);
              SCROLL.ABSOLUTE 8 LVALUE'
              SMIN
```

PF10/22

PF10/22 does the following:

- Scrolls the default window one half-page to the left side of the information in the stream it is displaying (As part of the SCROLL command, Session Manager locks the window.)
- Scrolls the LINE window one half-page to the left
- Unlocks the terminal keyboard
- Moves the LVALUE window over the word "LOCKED" on line 8 in the HEADER stream

The commands that define PF10/22 are:

```
CHANGE.PFK 10 'SCROLL.LEFT AMOUNT(HALF);
               SCROLL.LEFT LINE AMOUNT(HALF);
               CHANGE.TERMINAL CNTL(0);
               SCROLL.ABSOLUTE 8 LVALUE' SMIN
```

PF11/23

PF11/23 does the following:

- Scrolls the default window one half-page to the right side of the information in the stream it is displaying (As part of the SCROLL command, Session Manager locks the window.)
- Scrolls the LINE window one half-page to the right
- Unlocks the terminal keyboard
- Scrolls the LVALUE window over the word "LOCKED" on line 8 in the HEADER stream

The commands that define PF11/23 are:

```
CHANGE.PFK 11 'SCROLL.RIGHT AMOUNT(HALF);
               SCROLL.RIGHT LINE AMOUNT(HALF);
               CHANGE.TERMINAL CNTL(0);
               SCROLL.ABSOLUTE 8 LVALUE' SMIN
```

PF12/24

PF12/24 does the following:

- Scrolls the LINE window the maximum amount to the left (to column 1)
- Scrolls the default window the maximum amount to the left (to column 1)
- Moves the default window to the newest information in the stream and then unlocks the window
- Sets the maximum time that the keyboard can be locked to the last non-zero value entered
- Scrolls the LVALUE window over the word "UNLOCKED" on line 7 in the HEADER stream

The commands that define PF12/24 are:

```
CHANGE.PFK 12 'SCROLL.LEFT LINE AMOUNT(MAX);
               SCROLL.LEFT AMOUNT(MAX);
               UNLOCK.NEWEST;
               CHANGE.TERMINAL CNTL(LAST);
               SCROLL.ABSOLUTE 7 LVALUE'
               SMIN
```

Controlling the Terminal Keyboard

In order for Session Manager to update your display screen, TSO/E requires that the terminal keyboard remain locked. When the keyboard is locked, you cannot enter commands, and attention interruptions will not be processed. The CONTROL operand of the CHANGE.TERMINAL command lets you specify the number of seconds the keyboard is to be locked while a command is executing or while Session Manager is updating the screen. CONTROL can be any integer from 0 to 999.

If you want to wait for each TSO/E command or program to execute and have its output displayed before entering the next command, set CONTROL to a very high value. Note that attention interruptions will not be processed while the keyboard is locked.

If you want to enter TSO/E or Session Manager commands before the currently executing command has completed, set CONTROL to 0. The keyboard unlocks immediately after Session Manager rewrites the screen and you can then enter new commands. However, Session Manager only updates the screen after you have pressed the Enter key or any PF key. You must press the Enter key or a PF key every time you want new output from a command or program displayed.

Specifying CONTROL as a small number (10 to 20) is a compromise between the two extremes. A number in that range allows most TSO/E commands time to execute and have their output displayed, yet does not cause the terminal keyboard to be locked for long periods of time.

In the IBM-supplied default environment, the keyboard locks for a maximum of 15 seconds. The scrolling PF keys reset the CONTROL timer to 0 so that the keyboard immediately unlocks when you press one of them. PF key 12 (or 24) resets the CONTROL operand to the last non-zero value it had. This is done to allow you to scroll back and review previous information in a stream while a TSO/E command is executing without waiting for the command or program to complete.

Making a Copy of Your Display Screen

You can get a copy of the information on your display screen by using the SNAPSHOT command. This command places a copy of the information into a particular stream. You can then print the information by using the SMCOPY command. (See “Using TSO/E Commands” on page 184 for a description of SMCOPY.) An easy way to accomplish these steps is to define a PF key to issue the commands for you.

The following example illustrates how to define PF1 to issue the SNAPSHOT command placing a copy of the display screen in the EXTRA1 stream. The second command in the example defines PF4 to print the stream, and then erases the information in it so you can use the stream again. Notice that PF4 uses the PUT command to place the SMCOPY and SMPUT commands into the TSOIN stream.

```
CHANGE.PFK 1 'SNAPSHOT EXTRA1 FORMAT' SMIN
CHANGE.PFK 4 'PUT ''SMCOPY FROMSTREAM(EXTRA1) PRINT(A)
              PREFORMAT CAPS'' TSOIN; PUT ''SMPUT
              /CHANGE.STREAM EXTRA1 CLEAR/'' TSOIN' SMIN
```

Displaying Information About the Environment

Before you change the default environment, you might want to review how it is set up. You can display the characteristics of the default environment at your terminal by using the QUERY command. With this command, you can display the following information:

QUERY FUNCTION

Displays the following information for each currently defined session function:

- The name of the function
- The input, output, and copy streams for the function
- Whether the audible alarm is to sound when information enters the input and output streams for the function

- The intensity at which the information in the output and copy streams is to be displayed

QUERY PFK

Displays the following information for each currently defined program function (PF) key:

- The number of the PF key
- The name of the stream where the text-string is to be placed
- The identifier and delimiter characters used in the command
- The text-string used to define the key

QUERY STREAMS

Displays the following information for each currently defined stream:

- The name of the stream
- The numbers of the top and bottom lines in the stream
- The maximum size of the stream (in lines and bytes)
- The number of lines and bytes currently being used by the stream
- The type of stream it is (input, output, or extra)
- Whether the audible alarm is to sound when information enters the stream

QUERY TERMINAL

Displays the following information about the terminal environment:

- The control setting for the keyboard indicating the maximum time the keyboard is to remain locked
- Whether the audible alarm is to sound when the keyboard unlocks
- The current number of windows defined on the display screen
- The maximum number of windows that can be defined
- The name of the default window
- The location of the cursor
- The following information for each currently defined window:
  - The name of the window
  - The name of the stream that the window displays
  - Whether the window is locked
  - Whether you can enter data in the window
  - The name of the stream that is to receive the information entered in the window
  - The intensity at which the information in the stream is to be displayed
  - Whether the terminal's audible alarm is to sound when Session Manager scrolls the window to display new information in the stream
  - How long the window (when unlocked) is to be held in place before it is scrolled towards the bottom of the stream
  - How many lines of the window's old position are to be repeated when the window scrolls to the new position
  - How much new information must enter the stream before Session Manager updates the window

QUERY WINDOWS

Displays the following information for each currently defined window:

- The name of the window

- The starting location of the window on the display screen (in rows and columns)
- The size of the window (in lines and width)
- The name of the stream the window is displaying
- The numbers of the top and bottom lines of the stream that the window is presently displaying
- The numbers of the top and bottom lines of the stream that the window was displaying when it was last unlocked. These numbers are used when the UNLOCK.RESUME command is issued.
- The numbers of the top and bottom lines of the newest information in the stream

Saving The Environment

To help you easily manipulate your environment, Session Manager lets you save definitions on a stack, restore definitions from a stack, and reset the default environment via the RESET command.

Saving and Restoring Definitions

Session Manager maintains three push-down LIFO (last in, first out) stacks. The three kinds of stacks are *PF key stack*, *screen stack*, and *window stack*. You can use these stacks to save the current program function (PF) key definitions, screen definitions, and window definitions. For example, you might want to save the default PF key definitions, redefine the PF keys as TSO/E TEST subcommands for use with TSO/E TEST, and then restore the previous definitions when you leave TEST.

To place definitions in a stack, use the SAVE command. To remove or restore the definitions from the stack, use the RESTORE command. At the beginning of your terminal session, the stacks are empty. Issuing the RESTORE command then has no effect. After the first item has been saved on a stack, it remains as the first (bottom) item. It cannot be removed from the stack by issuing the RESTORE command although the definitions are restored to what they were. In effect, all items are removed except the first one.

PF Key Stack

When you issue the SAVE.PFK command, Session Manager saves all current PF key definitions as the top element on the PF key stack.

Screen Stack

When you issue the SAVE.SCREEN command, Session Manager saves the following information on the screen stack:

- A description of the display screen layout
- The location of the cursor
- The value indicating how long the keyboard is to remain locked while a command is executing (as set in the CHANGE.TERMINAL command) including the last non-zero control value entered
- The name of the default window
- The name and attributes of each currently defined window

Window Stack

When you issue the SAVE.WINDOW command, Session Manager saves the following information for the default window or for the window whose name is specified on the command:

- The audible alarm setting for the window
- The amount of time the window (when unlocked) is held in place before it is scrolled toward the bottom of the stream
- The number of lines from the window's old position that will be repeated when the window moves
- Whether data can be entered in the window

- The name of the stream that is to receive the information typed in the window and the intensity at which the information is to be displayed
- How often the window is to move over the new information in the stream
- The name of the stream the window is displaying
- The top and bottom line numbers in the stream that the window is currently displaying
- Whether the window is locked or unlocked

Resetting the Default Environment

If you accidentally deleted any of the windows on your display screen, you can use the RESET command to get the default display screen back. The RESET command lets you restart your Session Manager display environment. This command removes the entries from all of the stacks and re-executes the commands that created the default environment. The information in the streams is not altered.

The RESET command should not be followed by any other Session Manager command on the same line. A command entered on the same line will be executed before the RESET command can reestablish the default screen layout.

Ending Session Manager Support

To end Session Manager support of your TSO/E session, use the END command. END erases all information in your streams. Your TSO/E session then continues with standard display support. If you want Session Manager support again, you must reissue the TSO/E LOGON command.

Session Manager Processing

Listed below are some informational notes on the execution of Session Manager.

- Multiple TPUTs to a single line result in a single line for each TPUT. In addition, TSO/E does not guarantee the order of execution of multiple single line TPUTs generated from another address space (for example, the operator, another TSO/E user, or from a job you have submitted). Therefore, these multiple single line TPUTs might be out of order.
- All line-mode output produced when executing a full-screen program (in full-screen mode) is logged in the TSO/E function's output stream.
- All cross memory messages received when executing a full-screen program (in full-screen mode) are logged in the MSG function output stream.

Using TSO/E Commands

You can use any of the TSO/E commands or line-oriented functions with Session Manager. These commands can be found in *z/OS TSO/E Command Reference*. In addition, three TSO/E commands are provided for use with Session Manager: SMCOPY, SMFIND, and SMPUT. You can enter these commands from the keyboard, using PF keys, or via CLISTs. The functions of these commands are:

Command
Function

SMCOPY
Copies all or part of a stream or data set into another stream or data set (that is, stream to stream, stream to data set, data set to stream, or data set to data set). The receiving data set can be a SYSOUT data set.

SMFIND
Locates a string of characters in a Session Manager stream.

SMPUT
Places a string of characters in a Session Manager stream.

Using Command Procedures (CLISTs)

An easy way to execute a series of Session Manager or TSO/E commands is by using command procedures (CLISTs). A CLIST is an executable sequence of TSO/E and/or Session Manager commands, subcommands, or command procedure statements. You can use any TSO/E or Session Manager command in CLISTs. The same rules for entering Session Manager commands at your terminal apply when using a CLIST. (See "Entering Session Manager Commands" on page 167.)

This section describes several examples of CLISTs and how they can be used to modify the default environment. The CLIST in the first example redefines a program function (PF) key using Session Manager commands. The remaining examples show CLISTs that split your display screen horizontally and vertically. The split screen CLISTs are located in SYS1.ADFMAC1 so you do not have to type them in at your terminal. For a complete description of how to create, edit, invoke, and execute a CLIST, refer to *z/OS TSO/E CLISTs*.

Note: When you issue a WRITENR in a CLIST, the cursor is no longer positioned at the end of the text written to the screen. Instead, the cursor returns to the permanent cursor position to allow you to enter data.

Using CLISTs to Redefine Program Function (PF) Keys

Figure 13 on page 185 is an example of a simple CLIST that redefines PF9 using Session Manager commands. After the CLIST is executed, pressing PF9 causes the MAIN window to display the MESSAGE stream so you can see the messages from other TSO/E users, the operator, or background jobs. Pressing PF9 again repositions the MAIN window at its previous location and restores the previous PF9 definition. The steps in the CLIST are as follows:

1. Uses the SMPUT command to place the commands in the SMIN stream.
2. Saves the definitions of the existing PF keys on the PF key stack.
3. Changes the MSG function so that any messages from other TSO/E users, the operator, or background jobs are displayed in the message stream at normal intensity. The terminal's audible alarm is to sound when a message enters the stream.
4. PF9 then does the following:
   - Saves the definitions of the existing PF keys on the PF key stack
   - Saves the definition of the MAIN window on the window stack
   - Changes the MAIN window to display the MESSAGE stream where the messages have been placed
   - Redefines itself to:
     - Restore the MAIN window to display the stream it was previously displaying
     - Restore the PF key definitions. This resets PF9 to its previous definition

```
/*********************************************************************/
/* THIS CLIST SETS UP A SESSION MANAGER PROGRAM FUNCTION KEY.        */
/* IT ASSUMES THE IBM-SUPPLIED DEFAULT SCREEN LAYOUT IS THE CURRENT  */
/* SCREEN LAYOUT WHEN IT IS INVOKED:                                 */
/* - THE "MSG" FUNCTION IS CHANGED TO SEND MESSAGES FROM OTHER TSO/E */
/*   USERS TO THE "MESSAGE" STREAM, SOUNDING THE ALARM WHEN A        */
/*   MESSAGE IS RECEIVED.                                            */
/* - PF KEY 9 IS DEFINED TO MAKE THE "MAIN" WINDOW VIEW THE MESSAGE  */
/*   STREAM WHEN PRESSED. IF PRESSED AGAIN, PF 9 RETURNS THE "MAIN"  */
/*   WINDOW TO VIEWING THE STREAM IT WAS VIEWING WHEN PF 9 WAS FIRST */
/*   PRESSED.                                                        */
/* - THE EXISTING PF KEYS ARE SAVED BEFORE THE SCREEN IS MODIFIED.   */
/*********************************************************************/
                                                                   /* */
SMPUT /SAVE.PFK;+
       CHANGE.FUNCTION MSG OUTPUT(MESSAGE 1) ALARM(OUTPUT);+
       CHANGE.PFK 9 'SAVE.PFK;SAVE.WINDOW MAIN;+
                     CHANGE.WINDOW MAIN VIEW(MESSAGE);+
                     CHANGE.PFK 9 ''RESTORE.PFK;RESTORE.WINDOW MAIN'' +
                     SMIN' SMIN/ SMIN
```

Figure 13. A TSO/E CLIST that Redefines PF Key 9

Using CLISTs to Split the Display Screen

Often it is useful to view two streams at the same time. For example, you could view the listing of a program in the top window while using TSO/E TEST or some other debugging facility in the bottom window. You could also view HELP information or other data in the top window and enter new commands in the bottom window.

The examples on the following pages illustrate how you can use CLISTs to split the display screen horizontally and vertically. These CLISTs are located in SYS1.ADFMAC1 so you do not have to create them yourself. The ADFHSPLT CLIST splits the screen horizontally as shown in Figure 16 on page 188. The ADFVSPLT CLIST splits the screen vertically as shown in Figure 18 on page 190.

Both ADFHSPLT and ADFVSPLT depend on the ADFSETUP CLIST to place information in the HEADER and EXTRA2 streams. ADFSETUP requires a partitioned data set of CLISTs that you must allocate to the ddname of SYSPROC for implicit execution of the other CLISTS. If you use either ADFHSPLT or ADFVSPLT, execute ADFSETUP once at the beginning of your terminal session.

Note: If you issue the RESET command during your session, Session Manager clears the HEADER stream and then redefines it to contain only that information needed for the default environment. Therefore, you will need to re-execute the ADFSETUP CLIST to place the CLIST information in the HEADER stream again.

```
/*********************************************************************/
/* THIS CLIST PLACES DATA IN THE "HEADER" STREAM AND IN THE "EXTRA2" */
/* STREAM FOR THE "ADFHSPLT" AND THE "ADFVSPLT" CLISTS.              */
/*   IT SHOULD BE EXECUTED ONCE AT THE BEGINNING OF                  */
/* THE SESSION.  THIS CLIST REQUIRES A PDS CLIST DATA SET TO BE      */
/* ALLOCATED TO THE DDNAME OF "SYSPROC" FOR IMPLICIT EXECUTION OF    */
/* OTHER CLISTS - AS DESCRIBED IN "CLISTS: IMPLEMENTATION AND        */
/* REFERENCE".                                                       */
/*********************************************************************/
SMPUT /PUT 'TOP ===>  BOTTOM=>  RIGHT =>LEFT ==>' HEADER I(2)/
SET I=1
DO WHILE &I < (&SYSLTERM-5)
SMPUT /PUT | EXTRA2 I(2);PUT | EXTRA2 I(2); PUT | EXTRA2 I(2)/
SET &I=&I+1;
END
```

Figure 14. ADFSETUP CLIST

The comments in the ADFHSPLT CLIST (see Figure 15 on page 187) outline how this CLIST redefines the screen. Notice also how the LTITLE window is scrolled to the right and left to display "TOP" or "BOTTOM" in the HEADER stream. This indicates which window is the current default window so you can easily control which window Session Manager moves when you press one of the scroll PF keys. When you execute the ADFHSPLT CLIST, your display screen looks similar to Figure 16 on page 188.

```
PROC 0 LINE()
/*********************************************************************/
/* THIS CLIST SETS UP A "SPLIT - SCREEN" SESSION MANAGER SCREEN      */
/* LAYOUT. IT ASSUMES THE IBM - SUPPLIED DEFAULT SCREEN LAYOUT IS    */
/* THE CURRENT SCREEN LAYOUT WHEN IT IS INVOKED:                     */
/* - THE WINDOW "LINE" IS MOVED TO THE ROW SPECIFIED BY "LINE".      */
/* - THE WINDOW "CURRENT" IS EXPANDED TO FILL IN THE SPACE CREATED   */
/*       AND BECOMES THE "BOTTOM" WINDOW.                            */
/* - THE WINDOW "SPACE" IS DEFINED TO FILL IN THE AREA ABOVE THE     */
/*       SCROLL AMOUNT VALUE IF NEEDED.                              */
/* - THE SCREEN AND PF KEYS ARE SAVED BEFORE THE SCREEN IS MODIFIED. */
/*********************************************************************/
IF &LINE =     THEN SET &LINE = &EVAL((&SYSLTERM-5)/2)
IF &LINE < 2 | &LINE > &EVAL(&SYSLTERM-3) THEN +
 EXIT
SET &TOPS = &LINE -1
SET &BOT = &LINE +1
SET &BOTS = &EVAL(&SYSLTERM-1) - &BOT;
SET &BOTSX = &EVAL(&SYSLTERM-3) - &BOT;
IF &BOTSX > 0 THEN +
  SET &WIN = +
      DEF.W SPACE &BOT &EVAL(&SYSWTERM-17) &BOTSX 18 P(Y) V(EXTRA2);+
              S.A 2 SPACE;
ELSE +
  SET &WIN =
SMPUT /SAVE SCREEN;SAVE.PFK;+
       SAVE.WIN MAIN;SAVE.WIN LINE;SAVE.WIN CURRENT;+
       DEL.WIN MAIN;DEL.WIN LINE;DEL.WIN CURRENT;+
       DEFINE.WINDOW MAIN 1 1 &TOPS &EVAL(&SYSWTERM);+
       DEFINE.WINDOW LINE &LINE 1 1 &EVAL(&SYSWTERM);+
       DEFINE.WINDOW CURRENT &BOT 1 &BOTS &EVAL(&SYSWTERM-18);+
       &WIN;+
       RES.WIN CURRENT;RES.WIN LINE;RES.WIN MAIN;+
       CHANGE.WINDOW MAIN OVERLAP(1) HOLD(0)/
/*********************************************************************/
/* DEFINE PF 3/15 TO RE-INVOKE THIS CLIST, ACCEPTING AS INPUT THE    */
/* LINE (ROW) TO PLACE THE SPLIT AT. PRESSING PF 3/15 WITH NO INPUT  */
/* WILL RESTORE THE SCREEN AND PF KEYS TO THEIR ORIGINAL STATUS.     */
/*                                                                   */
/* PF 9/21 IS DEFINED TO SWITCH THE DEFAULT WINDOW BETWEEN THE "TOP" */
/* WINDOW AND THE "BOTTOM" WINDOW: THUS SETTING WHICH WINDOW IS      */
/* SCROLLED WHEN PF 5/17, 7/19, 8/20, 10/22, 11/23, 12/24 IS PRESSED.*/
/* THE "LTITLE" WINDOW IS MOVED TO DISPLAY THE NAME OF THE WINDOW    */
/* THAT IS CURRENTLY THE DEFAULT WINDOW. (THE NAMES WERE PREVIOUSLY  */
/* IN THE "HEADER" STREAM).                                          */
/*********************************************************************/
```

```
SMPUT /CHANGE.PF 3 'PUT ''RESTORE.SCREEN;RESTORE.PFKS;+
       RESTORE.PFKS;'' SMIN;+
       PUT ''%ADFHSPLT L(0¢1.)'' TSOIN' SMIN SUB(¢);+
       CHANGE.PF 9 'CHANGE TERMINAL DEFAULT(CURRENT);SAVE.PF;+
              SCROLL.LEFT LTITLE A(MAX);SCROLL.RIGHT LTITLE A(10);+
       CHANGE.PF 9 ''CHANGE.TERMINAL DEFAULT(MAIN);+
       RESTORE.PF;SCROLL.LEFT LTITLE A(MAX)'' SMIN' +
              SMIN;SCROLL.ABSOLUTE 11 LTITLE/
SMPUT /CHANGE.PF 15 'PUT ''RESTORE.SCREEN;RESTORE.PFKS;+
       RESTORE.PFKS;'' SMIN;+
       PUT ''%ADFHSPLT L(0¢1.)'' TSOIN' SMIN SUB(¢);+
       CHANGE.PF 21 'CHANGE TERMINAL DEFAULT(CURRENT);SAVE.PF;+
              SCROLL.LEFT LTITLE A(MAX);SCROLL.RIGHT LTITLE A(10);+
       CHANGE.PF 21 ''CHANGE.TERMINAL DEFAULT(MAIN);+
       RESTORE.PF;SCROLL.LEFT LTITLE A(MAX)'' SMIN' +
              SMIN;SCROLL.ABSOLUTE 11 LTITLE/
```

Figure 15. ADFHSPLT CLIST

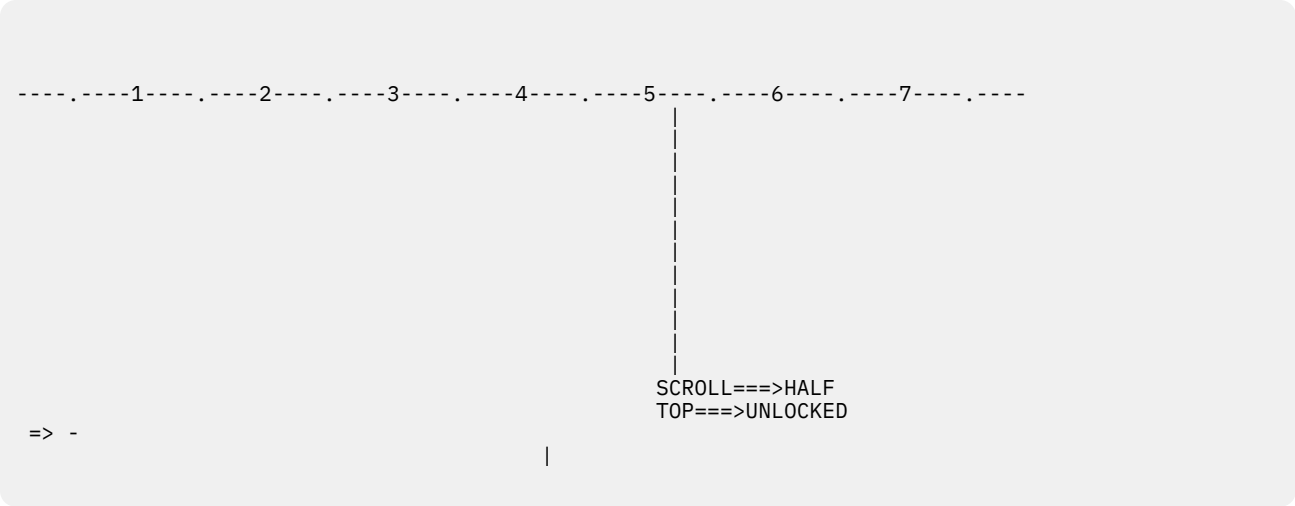


Figure 16. Horizontal Split of the Display Screen

The ADFVSPLT CLIST is very similar to the ADFHSPLT CLIST. You might find this CLIST useful when testing programs or comparing members of a partitioned data set. For example, you could use the CLIST to do the following:

- Use the LISTDS command to list the members of a partitioned data set
- Lock the MAIN window by pressing PF10 (scrolling to the left when the window is already at the left border merely locks the window)
- Execute the ADFVSPLT CLIST and press PF9 to make the RIGHT window the default window
- Press PF12 to unlock the RIGHT window and enter the LISTDS command for the second data set whose members you want to display
- By using PF9 to change the default window to the left side (the MAIN window), you can use the other PF keys to scroll the windows and compare the members of the data sets

Figure 17 on page 189 is the ADFVSPLT CLIST and Figure 18 on page 190 illustrates what the display screen looks like when this CLIST has been executed.

```
PROC 0 COL()
/*********************************************************************/
/* THIS CLIST SETS UP A "SPLIT - SCREEN" SESSION MANAGER SCREEN      */
/* LAYOUT. IT ASSUMES THE IBM - SUPPLIED DEFAULT SCREEN LAYOUT IS    */
/* THE CURRENT SCREEN LAYOUT WHEN IT IS INVOKED:                     */
/* - THE "MAIN" WINDOW IS SPLIT VERTICALLY:                          */
/* - THE "MAIN" WINDOW IS REDEFINED WITH A NEW WIDTH OF "COL" -1     */
/*       AND BECOMES THE "LEFT" WINDOW.                              */
/* - A NEW WINDOW "SPLIT" IS CREATED TO PROVIDE A VISUAL INDICATION  */
/*   OF THE SPLIT. IT VIEWS THE "EXTRA2" STREAM IN WHICH VERTICAL    */
/*   BARS HAVE BEEN PLACED PREVIOUSLY.                               */
/* - THE WINDOW "RIGHT" IS DEFINED TO FILL IN THE AREA TO THE RIGHT  */
/*   OF THE "SPLIT" WINDOW. ITS ATTRIBUTES ARE FILLED IN BY          */
/*   RESTORING A COPY OF THE ATTRIBUTES OF THE "MAIN" ("RIGHT")      */
/*   WINDOW FROM THE "WINDOW" STACK.                                 */
/* - THE SCREEN AND PF KEYS ARE SAVED BEFORE THE SCREEN IS MODIFIED. */
/*********************************************************************/
IF &COL =  THEN SET &COL = &EVAL((&SYSWTERM)/2)
IF &COL < 5 | &COL > &EVAL(&SYSWTERM-5) THEN +
 EXIT
SET &LEFTS = &COL - 1
SET &RIGHT = &COL + 2
SET &RIGHTS = &EVAL(&SYSWTERM+1) - &RIGHT;
SMPUT /SAVE.SCREEN;SAVE.PFK;+
       SAVE.WIN MAIN;SAVE.WIN MAIN;DELETE.WIN MAIN;+
       DEFINE.WIN MAIN 1 1 &EVAL(&SYSLTERM-5) &LEFTS;+
       DEFINE.WIN SPLIT 1 &COL &EVAL(&SYSLTERM-5) 2 +
             HOLD(0) VIEW(EXTRA2) PROTECT(YES);+
             SCROLL.ABSOLUTE 2 SPLIT;+
       DEFINE.WIN RIGHT 1 &RIGHT &EVAL(&SYSLTERM-5) &RIGHTS;+
       RESTORE.WIN MAIN;RESTORE.WIN RIGHT/
/*********************************************************************/
/* PF 9/21 IS DEFINED TO SWITCH THE DEFAULT WINDOW BETWEEN THE "LEFT"*/
/* WINDOW AND THE "RIGHT" WINDOW: THUS SETTING WHICH WINDOW IS       */
/* SCROLLED WHEN PF 5/17, 7/19, 8/20, 10/22, 11/23, 12/24 IS PRESSED.*/
/* THE "LTITLE" WINDOW IS MOVED TO DISPLAY THE NAME OF THE WINDOW    */
/* THAT IS CURRENTLY THE DEFAULT WINDOW. (THE NAMES WERE PREVIOUSLY  */
/* IN THE "HEADER" STREAM.                                           */
/*                                                                   */
/* DEFINE PF 3/15 TO RE-INVOKE THIS CLIST, ACCEPTING AS INPUT THE    */
/* LINE (ROW) TO PLACE THE SPLIT AT. PRESSING PF 3/15 WITH NO INPUT  */
/* WILL RESTORE THE SCREEN AND PF KEYS TO THEIR ORIGINAL STATUS.     */
/*********************************************************************/
```

```
SMPUT /CHANGE.PF 9 'CHANGE.TERMINAL DEFAULT(RIGHT);SAVE.PF;+
              SCROLL.LEFT LTITLE A(MAX);SCROLL.RIGHT LTITLE A(20);+
       CHANGE.PF 9 ''CHANGE.TERMINAL DEFAULT(MAIN);RESTORE.PF;+
              SCROLL.LEFT LTITLE A(MAX);+
              SCROLL.RIGHT LTITLE A(28)'' SMIN' SMIN;+
              SCROLL.ABSOLUTE 11 LTITLE;+
              SCROLL.RIGHT LTITLE A(28);+
       CHANGE.PF 3 'PUT ''RESTORE.SCREEN;RESTORE.PF;+
              RESTORE.PF'' SMIN;+
              PUT ''%ADFVSPLT COL(0¢1.)'' TSOIN' SMIN SUB(¢)/
SMPUT /CHANGE.PF 21 'CHANGE.TERMINAL DEFAULT(RIGHT);SAVE.PF;+
              SCROLL.LEFT LTITLE A(MAX);SCROLL.RIGHT LTITLE A(20);+
       CHANGE.PF 21 ''CHANGE.TERMINAL DEFAULT(MAIN);RESTORE.PF;+
              SCROLL.LEFT LTITLE A(MAX);+
              SCROLL.RIGHT LTITLE A(28)'' SMIN' SMIN;+
              SCROLL.ABSOLUTE 11 LTITLE;+
       CHANGE.PF 15 'PUT ''RESTORE.SCREEN;RESTORE.PF;+
              RESTORE.PF'' SMIN;+
              PUT ''%ADFVSPLT COL(0¢1.)'' TSOIN' SMIN SUB(¢)/
```

Figure 17. ADFVSPLT CLIST

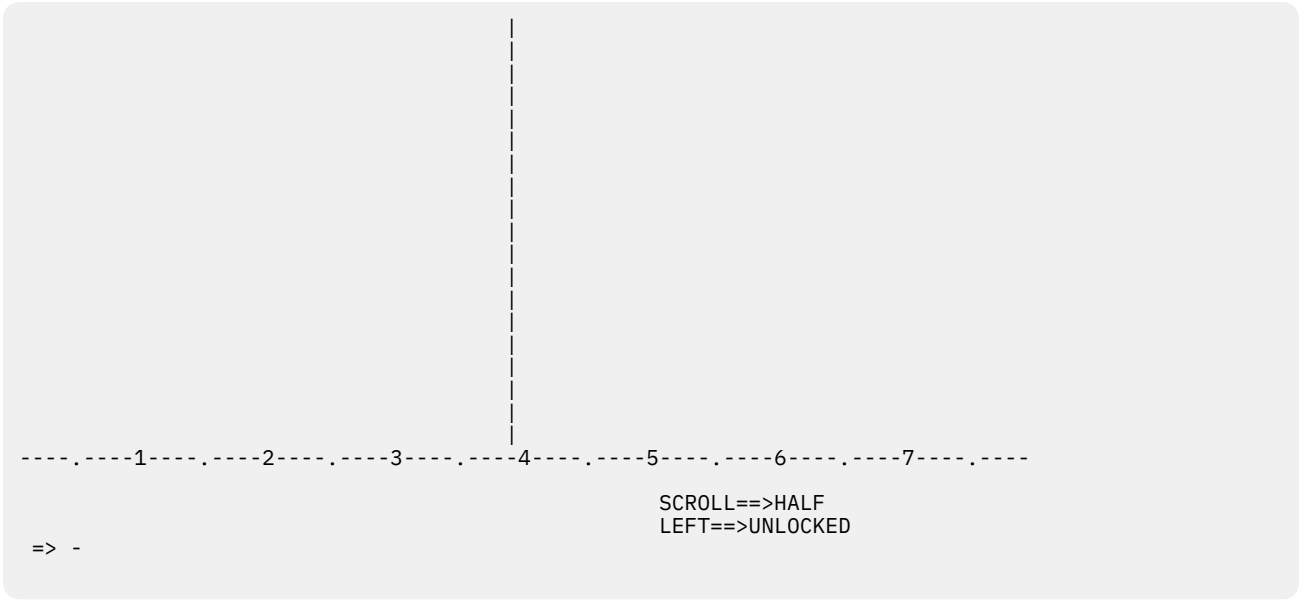


Figure 18. Vertical Split of the Display Screen

Both ADFHSPLT and ADFVSPLT define PF 3/15 as the END key. When you are ready to go back to your original screen, just press PF3/15. If at any time you want to switch back to the default screen layout and PF key definitions, clear the screen by pressing the CLEAR key and type in the RESET command.

Appendix A. Full-Screen Logon Processing

Full-screen logon panels are available to an IBM 3270 terminal user if the terminal display format is 80 columns wide by 24 or more rows long. After you enter LOGON *user_id*, TSO/E displays a panel with the logon operand fields relevant to your session. The operand values you entered with the LOGON command are displayed on the panel, except for the password.

When you log on with your user ID as the only operand, other logon operand values default to the values present on the panel the last time you logged on. You can override the default values by entering operands with the LOGON command or by typing over the default values on the logon panel.

If you are a RACF-defined user, you can change your current password by entering it in the PASSWORD field and by entering the value that is to replace the current password in the NEW PASSWORD field. The PASSWORD fields are nondisplay fields.

Also, if you are a RACF-defined user, you can change the security label shown in the SECLABEL field on the logon panel by typing over the value initially displayed on the panel.

When you are satisfied with the values on the logon panel, press the Enter key to process the LOGON command. To process a full-screen panel, you must supply TSO/E with at least a valid user ID. Also, TSO/E will not process a panel until your other operand values (for example, your password) are acceptable.

Note: In the following discussion, *yourid* represents your TSO/E user ID and the logon panel values are hypothetical.

Command Entry Field

In the COMMAND field of the logon panel, you can enter a single command of up to 80 characters long.

Note:

1. TSO/E executes the command you entered *after* any command the TSO/E administrator entered in the PARM field on the EXEC statement of the logon procedure. This would be followed by any commands found on the data stack.
2. TSO/E does *not* execute the command you enter in the COMMAND field of the logon panel if the command specified in the PARM field of the logon procedure fails. For example, the command you enter fails if TSO/E cannot find the command in the PARM field.

Full-Screen Logon For a Non-RACF User

If you are a non-RACF-defined user:

1. Enter LOGON *yourid*. TSO/E displays the following panel with default values:

© Copyright IBM Corp. 1988, 2024

```
------------------------------ TSO/E LOGON ------------------------------
PF1/PF13 ==> Help    PF3/PF15 ==> Logoff    PA1 ==> Attention    PA2 ==> Reshow
You may request specific HELP information by entering a '?' in any entry field.
   ENTER LOGON PARAMETERS BELOW:

   USERID    ===> YOURID

   PASSWORD  ===> -

   PROCEDURE ===> MYPROC

   ACCT NMBR ===> 00123

   SIZE      ===> 5800

   PERFORM   ===>

   COMMAND   ===>


   ENTER AN 'S' BEFORE EACH OPTION DESIRED BELOW:

          -NOMAIL         -NONOTICE        -RECONNECT
```

2. Type in your password in the password field. For your protection, TSO/E inhibits the display of the password on the screen. You can change any value on the panel, except your user ID. To change a value, use the cursor positioning keys to position the cursor in the appropriate field.
3. Press the Enter key to process your input. After TSO/E verifies your logon values, it displays the following message:

```
YOURID LOGON IN PROGRESS AT 9:00 ON 2/7/88
```

4. When you have successfully logged on, TSO/E displays system messages, if any, sometimes followed by three asterisks, ***.

   Whenever you see three asterisks, press the Enter key to continue.

Full-Screen Logon for a RACF-Defined User

If you are a RACF-defined user, and this is your first logon using the full-screen logon panel:

1. Enter LOGON *yourid*. TSO/E displays the following panel with default values:

```
------------------------------ TSO/E LOGON ------------------------------
PF1/PF13 ==> Help    PF3/PF15 ==> Logoff    PA1 ==> Attention    PA2 ==> Reshow
You may request specific HELP information by entering a '?' in any entry field.
   ENTER LOGON PARAMETERS BELOW:                RACF LOGON PARAMETERS:

   USERID    ===> YOURID                        SECLABEL     ===>

   PASSWORD  ===> -                             NEW PASSWORD ===>

   PROCEDURE ===> MYPROC                        GROUP IDENT  ===>

   ACCT NMBR ===> 00123

   SIZE      ===> 5800

   PERFORM   ===>

   COMMAND   ===> EXEC (SETUP)


   ENTER AN 'S' BEFORE EACH OPTION DESIRED BELOW:

          -NOMAIL         -NONOTICE        -RECONNECT         -OIDCARD
```

2. Enter your password (commonly a group password) in the PASSWORD field. If you want to change your password, enter the new password in the NEW PASSWORD field. You may also enter other logon operand values.
3. Press the Enter key to enter your input to TSO/E. After preliminary messages, TSO/E generally displays three asterisks, ***. Whenever you see ***, press the Enter key to continue. You then see the READY message:

```
READY
```

   If you specified a command in the COMMAND field of the logon panel, in this case, EXEC (SETUP), TSO/E processes your command. Had you left the COMMAND field blank, you could begin entering commands immediately after the READY message.

Notes:

1. **Note:** The SECLABEL field on the full-screen logon panel is displayed only if your installation is using security labels.

   When you log on to TSO/E using the full-screen logon panel, the values that are shown in some of the fields, such as PROCEDURE, ACCT NMBR, and SECLABEL are the values that you entered for your previous TSO/E session. You can choose a different value by typing in a new value in the field. The new value that you enter is then saved and is displayed whenever you log on again until you either enter a different value or blank out the field.
2. If PasswordPrePrompt is active, you are prompted for your password before the display of the full-screen panel.

The values that you are required to enter on the logon panel depend on your installation. See your TSO/E administrator for logon instructions.

Error Prompting

Full-screen logon processing checks the syntax of each input field. When TSO/E discovers a syntax violation, it:

- Sounds an alarm (if the terminal is so equipped)
- Displays an appropriate syntax message on the third line of the panel
- Locks the fields of correct entries
- Places an asterisk (*) before the syntactically incorrect entry or entries
- Highlights syntactically incorrect entry or entries
- Positions the cursor in the first syntactically incorrect entry field.

TSO/E highlights nonsyntactical errors in the same ways. For example, if on the LOGON command, you enter an incorrect user ID that is syntactically valid, TSO/E displays a panel containing any parameter values you entered on the LOGON command, including the incorrect user ID. If RACF is installed and active, TSO/E also displays RACF entry fields. No default values are available, however.

Note: If you have an APL keyboard and your terminal is in APL mode, the logon fields do not respond predictably to information you enter. Therefore, it is best to complete your logon and then switch to APL mode.

Program Function Key Support for Full-Screen Logon

During full-screen logon processing, TSO/E processes interrupts caused only by the following keys.

PA1 Attention Key - Full-screen logon attention processing erases the present screen. When the word LOGON appears at the top of the blank screen, you may enter LOGOFF or enter the LOGON command again. If you enter anything else, TSO/E issues an error message.

PA2 Reshow Key - When you modify entries on a panel and then decide to start over, press PA2 to retrieve and re-display the entries you have modified. TSO/E displays the logon information that was on the screen when you last caused an interrupt.

PF1/PF13 HELP Key - Whenever you enter an incorrect value, TSO/E prompts you for a correct one. At this time, you can press either PF1 or PF13 to request syntactical and functional information about the entry field. TSO/E displays the appropriate text and, except for the password entry fields, also displays the incorrect data.

Note: Whenever you can enter data in the USERID, PASSWORD, or RACF PASSWORD field, pressing PF1 or PF13 requests information on *all* of the logon operands. In this case, when you are finished reading a panel of information, press the Enter key to notify TSO/E that you are ready to view the next panel. Any parameter values you have entered are also displayed with the corresponding HELP text.

Question Mark (?) HELP Key - You can enter a question mark (?) in any entry field to obtain syntactical and functional information about an entry. When you enter a question mark in an entry field containing data, TSO/E erases it. In addition, whenever you can enter data in the USERID, PASSWORD, or RACF PASSWORD field, you may enter a question mark for any entry to obtain information about the entry(ies). If you enter more than one question mark, use the Enter key to request the next panel of information.

PF3/PF15 End HELP Key - Press PF3 or PF15 to notify TSO/E that you want to stop viewing the information you requested using PF1 or PF13. If TSO/E is displaying HELP information for one field only, you may also use the Enter key or any PF key to end the HELP display and return to the logon panel.

PF3/PF15 LOGOFF Key - Any time TSO/E is displaying the logon panel, you may press either PF3 or PF15 to log off.

Appendix B. Using Line Mode Edit

To enter information in data sets, you can use the EDIT command in the following ways:

- To create a new data set and begin entering data, use the NEW operand and specify the data set name. If you specify a member name, the system creates a partitioned data set. Otherwise the system creates a sequential data set.
- To specify the initial mode of entry, use either the EMODE (edit mode) or IMODE (input mode) operands. EMODE is the default for existing data sets and IMODE is the default for new data sets.
- If you were editing a data set right before a system failure, you can recover data by issuing the EDIT command with the name of the data set you were editing and include the RECOVER operand.
- To edit a data set with a particular format or in a particular programming language, use one of the following operands:
  - ASM - for assembler language statements
  - COBOL - for COBOL statements
  - CLIST - for CLIST statements
  - CNTL - for JCL statements
  - TEXT - for text that consists of both uppercase and lowercase characters
  - DATA - for text that can be retrieved or used as input data

Modes of Operation

The EDIT command has two modes of operation: input mode and edit mode. You enter data into a data set when you are in input mode. You enter subcommands and their operands when you are in edit mode.

You must specify a data set name when you enter the EDIT command. If you specify the NEW operand, the system places you in input mode. If you do not specify the NEW operand, you are placed in edit mode if your specified data set is not empty. If the data set is empty, you are placed in input mode.

If you have limited access to your data set, by assigning a password, you can enter a slash (/) followed by the password of your choice after the data set name operand of the EDIT command.

Entering either EMODE or IMODE operands on the EDIT command overrides the normal mode setting described above. The specification of the RECOVER operand on the EDIT command places you in edit mode upon recovery. Refer to “Recovering an EDIT Work File” on page 201 for more information about the RECOVER operand.

Input Mode

In input mode, you type a line of data and then enter it into the data set by pressing the Enter key. You can enter lines of data as long as you are in input mode. One typed line of input becomes one record in the data set.

Note: If you enter a command or subcommand while you are in input mode, the system adds it to the data set as input data. Enter a null line to return to edit mode before entering any subcommands.

Line Numbers

Unless you specify otherwise, the system assigns a line number to each line as it is entered. The default is an interval of 10. Line numbers make editing much easier, because you can refer to each line by its own number.

© Copyright IBM Corp. 1988, 2024

Each line number consists of up to eight digits, with the significant digits justified on the right and preceded by zeros. Line numbers are placed at the beginning of variable-length records and at the end of fixed-length records. (Exception: Line numbers for COBOL fixed-length records are placed in the first six positions at the beginning of the record.) When you are working with a data set that has line numbers, you can have the new line number listed at the start of each new input line. If you are creating a data set without line numbers, you can request that a prompting character be displayed at the terminal before each line is entered. Otherwise, none is issued.

Input records are converted to uppercase characters, unless you specify the ASIS or TEXT operand. The TEXT operand also specifies that character-deleting indicators and tabulation characters are recognized, but all other characters are added to the data set unchanged.

All Assembler source data sets must consist of fixed-length records, 80 characters long. These records might not be line numbers. If the records are line-numbered, the number can be located anywhere within columns 73 to 80 of the stored record (the printed line number always appears at the left margin).

Syntax Checking

You can have each line of input checked for proper syntax. The system checks the syntax of statements for data sets having FORT descriptive qualifiers. Input lines are collected within the system until a complete statement is available for checking.

When an error is found during syntax checking, an appropriate error message is issued and edit mode is entered. You can then take corrective action, using the subcommands. When you want to resume input operations, press the Enter key. Input mode is then entered and you can continue where you left off. Whenever statements are being checked for syntax during input mode, the system prompts you for each line to be entered unless you specify the NOPROMPT operand for the INPUT subcommand.

Continuation of a Line in Input Mode

In input mode, there are two independent situations that require you to indicate the continuation of a line by ending it with a hyphen or plus sign (that is, a hyphen or plus sign followed immediately by pressing the Enter key). The situations are:

- The syntax checking facility is being used.
- The data set type is CLIST (variable-length records).

If none of these situations apply, avoid ending a line with a hyphen (minus sign) because it is removed by the system before storing the line in your data set.

You must use the hyphen when the syntax checking facility is active to indicate that the logical line to be syntax checked consists of multiple input lines. The editor then collects these lines (removing the hyphens) and passes them as one logical line to the syntax scanner. However, each individual input line (with its hyphen removed) is also stored separately in your data set.

The hyphen is used to indicate logical line continuation in CLISTs. If the CLIST is in variable-length record format (the default), the hyphen is not removed by EDIT, but becomes part of the stored line in your data set and is recognized when executed by the EXEC command processor. If the CLIST is in fixed-length record format, a hyphen, placed eight character positions before the end of the record and followed by a blank, is recognized as a continuation when executed by the EXEC command processor. This assumes that the line number field is defined to occupy the last eight positions of the stored record. For example, if the operand LINE(80) is specified on the EDIT command when defining the CLIST data set, the hyphen must be placed in data position 72 of the input line followed immediately by a blank. Location of a particular input data column is described under the TABSET subcommand of EDIT.

Note that these rules apply only when entering data in input mode. When you use a subcommand (for example, CHANGE or INSERT) to enter data, a hyphen at the end of the line indicates subcommand continuation. The system appends the continuation data to the subcommand.

INPUT Mode in the Background

When the EDIT command is executed in the background and input mode is requested, blank lines should not be entered into the data set. EDIT interprets a blank line as a null line causing a switch from input mode to edit mode. When it is necessary to incorporate blank lines into your data set, certain methods can be followed. One method is to insert an unused character string wherever a blank line is required. Before ending the edit session, insert the CHANGE subcommand to change the character string to blanks. Figure 19 on page 197 illustrates how this is done:

```
EDIT   examp4.cntl  NEW
INPUT
00010 LOGON user4 PROC(proca)
00020 PROFILE PREFIX(userID)
00030 EDIT p.data NEW
00040 line one
00050 @@@@@
00060 line two
00070 @@@@@
00080 line three
00090 @@@@@
00100 line four
00110
00120 c 10 999 /@@@@@/ / all
00130 list
00140 end save
00150 (null line)
end save
READY
SUBMIT examp4.cntl NOTIFY JOBCHAR(a)
JOB USER4A(JOB00001) SUBMITTED
READY
```

Figure 19. Entering Blank Lines Into Your Data Set

An alternate method is to specify the operand EMODE on the EDIT command that is to be executed in the background. With this method, each new line of data is preceded by a line number if the data set has line numbers.

To insert a line of data ending in a hyphen in situations where the system would remove the hyphen (that is, while in subcommand mode or in input mode for other than a CLIST data set), enter a hyphen in the next-to-last column, a blank in the last column, and immediately press the Enter key.

Creating a Data Set

When creating a data set, you must first request input mode. You can do this by entering one of the following:

- The NEW operand on the EDIT command
- The IMODE operand on the EDIT command
- The INPUT subcommand while in edit mode
- The INSERT subcommand with no operands, while in edit mode
- A null line, if the system is in edit mode

After you enter the EDIT command with either the NEW or IMODE operands, the system displays the following message:

```
INPUT
```

For example:

Operation: Add data to an existing data set using the IMODE operand.

Known: To add data, you want to go into input mode immediately.

Enter:

```
EDIT cmdproc.clist IMODE
```

Edit Mode

You can enter subcommands to edit data sets when you are in edit mode. You can edit data sets that have line numbers by referring to the number of the line that you want to edit. This is called *line-number editing*. You can also edit data by referring to specific items of text within the lines. This is called *context editing*. A data set having no line numbers can be edited only by context. Context editing is performed by using subcommands that refer to the current line value or a character combination, such as with the FIND or CHANGE subcommands. There is a pointer within the system that points to a line within the data set. Normally, this pointer points to the last line that you referred to. You can use subcommands to change the pointer so that it points to any line of data that you choose. You can then refer to the line that it points to by specifying an asterisk (*) instead of a line number. Table 6 on page 198 shows where the pointer points at completion of each subcommand.

Note: A current-line pointer value of zero refers to the position before the first record, if the data set does not contain a record zero.

When you edit data sets with line numbers, the line number field is not involved in any modifications made to the record except during renumbering. Also, the only editing operations that is performed across record boundaries is the CHANGE and FIND subcommands, when the TEXT and NONUM operands have been specified for the EDIT command. In CHANGE and FIND, an editing operation is performed across only one record boundary at a time.

Table 6. How EDIT Subcommands Affect the Line Pointer Value

EDIT Subcommands	Description	Pointer Value at Completion
ALLOCATE	Allocates data sets and file names.	No change
ATTRIB	Builds a list of attributes for non-VSAM data sets.	No change
BOTTOM	Moves the pointer to the last record in the data set.	Last line (or zero for empty data sets)
CHANGE	Alters the contents of a data set.	Last line changed
CKPT	Protects input or modifications to a data set.	No change
COPY	Copies records within the data set.	Last line copied
DELETE	Removes records.	Line preceding deleted line (or zero if the first line of the data set has been deleted)
DOWN	Moves the pointer toward the end of the data.	Line n relative lines below the last line referred to, where n is the value of the count operand, or bottom of the data set (or line zero for empty data sets)
END	Terminates the EDIT command.	No change
EXEC	Executes a CLIST or REXX exec.	No change

Table 6. How EDIT Subcommands Affect the Line Pointer Value (continued)

EDIT Subcommands	Description	Pointer Value at Completion
FIND	Locates a character string.	Line containing specified string, if any; else, no change
FORMAT (part of a program product)	Formats and lists data.	No change
FREE	Releases previously allocated data sets.	No change
HELP	Explains available subcommands.	No change
INPUT	Prepares the system for data input.	Last line entered
INSERT	Inserts records.	Last line entered
Insert/Replace/Delete	Inserts, replaces, or deletes a line.	Inserted line or replaced line or line preceding the deleted line if any (or zero, if no preceding line exists).
LIST	Prints out specific lines of data.	Last line listed
MERGE (part of a program product)	Combines all or parts of data sets.	Last line
MOVE	Moves records within a data set.	Last line moved
PROFILE	Specifies characteristics of your user profile.	No change
RENUM	Numbers or renumbers lines of data.	Same relative line
RUN	Causes compilation and execution of data sets.	No change
SAVE	Retains the data set.	No change or same relative line
SCAN	Controls syntax checking.	Last line scanned, if any
SEND	Allows you to communicate with the system operator and with other users.	No change
SUBMIT	Submits a job for execution in the background.	No change
TABSET	Sets the tabs.	No change
TOP	Sets the pointer to zero value.	Zero value
UNNUM	Removes line numbers from records.	Same relative line
UP	Moves the pointer toward the start of the data set.	Line n relative lines above the last line referred to, where n is the value of the count operand, (or line zero for empty data sets).

Table 6. How EDIT Subcommands Affect the Line Pointer Value (continued)		
EDIT Subcommands	**Description**	**Pointer Value at Completion**
VERIFY	Causes current line to be listed whenever the current line pointer changes or the text of the current line is modified.	No change

Changing from One Mode to Another

If you specify an existing data set name as an operand for the EDIT command, you begin processing in edit mode. If you specify a new data set name or an old data set with no records as an operand for the EDIT command, you begin processing in input mode.

You change from edit mode to input mode when:

- You enter the INPUT subcommand.
- You press the Enter key before typing anything.
- You enter the INSERT subcommand with no operands.

If this is the first time during your current usage of EDIT that input mode is entered, input begins at the line after the last line of the data set for data sets which are not empty, or at the first line of the data set for empty data sets. If this is not the first time during your current usage of EDIT that input mode is entered, input begins at the point following the data entered when last in input mode.

If you use the INPUT subcommand without the R operand and the line is null (that is, it contains no data), input begins at the specified line. If the specified line contains data, input begins at the first increment past that line. If you use the INPUT subcommand with the R operand, input begins at the specified line, replacing existing data, if any.

You switch from input mode to edit mode when:

- You press the Enter key before typing anything.
- You cause an attention interruption.
- There is no more space for records to be inserted into the data set and re-sequencing is not allowed.
- An error is discovered by the syntax checker.

Tabulation Characters

When you enter the EDIT command into the system, the system establishes a list of tab setting values for you, depending on the data set type. Logical tab setting values might not represent the actual tab setting on your terminal. You can establish your own tab settings for input by using the TABSET subcommand. A list of the default tab setting values for each data set type is presented in the TABSET subcommand description. The system scans each input line for tabulation characters produced by pressing the TAB key on the terminal. The system replaces each tabulation character by as many blanks as are necessary to position the next character at the appropriate logical tab setting.

When tab settings are not in use, each tabulation character encountered in all input data is replaced by a single blank. You can also use the tabulation character to separate subcommands from their operands.

Executing User-Written Programs

You can compile and execute the source statements contained in certain data set types by using the RUN subcommand. The RUN subcommand makes use of optional program products. The specific requirements are discussed in the description of the RUN command.

Terminating the EDIT Command

You can terminate the EDIT operation at any time by switching to edit mode (if you are not already in edit mode) and entering the END subcommand. Before terminating the EDIT command, you should be sure to store all data that you want to save. You can use the SAVE subcommand or the SAVE operand of the END subcommand for this purpose.

Recovering an EDIT Work File

In the event of an abnormal termination, the recovery facility of EDIT enables you to recover changes or modifications made during an edit session (applicable in foreground only). To recover the work file after an abnormal termination has occurred during an edit session, the EDIT command should be reissued specifying the RECOVER operand, along with any other operands specified initially. This facility is optional to both the installation and/or the TSO user.

Certain specifications must be met before a work file becomes recoverable. They are:

- The installation must not have specified the NORECOVER attribute to your user ID. If the NORECOVER attribute was assigned, the data set is not recoverable.
- To be recoverable, you must enter the PROFILE command containing the RECOVER operand prior to the edit session.

If the conditions above are met, EDIT creates a work file and updates it while your edit session progresses. If the edit session terminates normally, the work file is deleted immediately upon termination. If the edit session is terminated abnormally, the work file is kept and made available at the beginning of your next edit session.

Checkpointing a Data Set

The CKPOINT subcommand of EDIT gives you the ability to automatically checkpoint a data set during the input or modification phase of the edit session. The invocation of checkpointing is controlled through the use of the CKPOINT subcommand. See the CKPOINT subcommand of EDIT for the syntax description.

Recovering After a System Failure

To recover data from your last edit session, issue the EDIT command entering the same data set name that you were working on at the time of the failure and include the RECOVER operand. You are placed in edit mode and the work file data set is used as input for the current edit session. The current line pointer is positioned at the top of the data set.

Note the following:

- If you specify IMODE upon re-entering your edit session, or if you give a data set disposition of NEW, the recovery feature always puts your session in edit mode.
- If the RECOVER operand is not specified, you are prompted and given a choice of RECOVER or NORECOVER.
- If the RECOVER operand is specified and the work file data set name does not match the edited data set name, an error message is issued. You are prompted and given a choice of recovering or not recovering the data set.
- If the RECOVER operand is specified and the work file data set does not exist, an error message is issued.

The example shown in Figure 20 on page 202 illustrates the different stages of an edit session and the actions necessary to recover it.

```
READY
PROFILE RECOVER
READY
EDIT lions old data
EDIT
ckpoint 5
list
00010 THE
00020 EDIT
00030 LOST,
00040 REENTER
00050 COMMAND
00060 AND
00070 SAVE
00080 ENTRY
c 30 /lost,/recovery/
c 40 /reenter/feature/
c 50 /command/saves/
c 60 /and/you/
c 70 /save/time and/
     (System automatically takes a checkpoint after
     fifth line of modifications.)
c 80 /entry/repetition/
     (Assuming system failure has occurred here, your edit
     session will terminate abnormally.  When the system
     is restored, issue the LOGON command and reenter the
     EDIT command including the RECOVER operand.)

EDIT lions old data RECOVER
EDIT
list
00010 THE
00020 EDIT
00030 RECOVERY
00040 FEATURE
00050 SAVES
00060 YOU
00070 TIME AND
00080 ENTRY
c 80 /entry/repetition/
(Note:  The last line was not kept.  All other changes
     were kept in the EDIT work file (utility data set)
     making it necessary to reenter only one line.)
```

Figure 20. Sample Edit Session Using the CKPOINT Subcommand and the RECOVER Operand of EDIT

Recovering After an Abend

When an abend occurs after issuing the SAVE subcommand of EDIT because there is not enough space (B37, D37, E37) in your data set or on the volume in which your data set resides, message IKJ52432A is issued. Termination does not occur, even if all attempts to save the data set are unsuccessful. You can respond to the system prompt with one of the following options:

- Enter the SAVE subcommand specifying a different data set name.
- Enter RETAIN to terminate your edit session. The EDIT work file (utility data set) is checkpointed and retained. Recovery is possible at the beginning of your next edit session.
- Enter END to terminate your edit session. With this option, the EDIT work file is not available for recovery at your next edit session.
- Entering any other valid subcommand of EDIT at this time causes the abend to be disregarded and your edit session continues.

Using the RETAIN option allows you to end your edit session and then perform any space recovery measures necessary to obtain additional space. The RECOVER operand on the EDIT command can be used to recover your data set during your next edit session. Refer to “Recovering After a System Failure” on page 201 for the correct procedure.

When your edit session is terminated by a system, operator, or time allocation (abend code X22), the EDIT work file is checkpointed and retained, if any modifications were made. This allows you to invoke EDIT's recovery feature after your next logon is issued. For any other abends, you are prompted for END or SAVE through a message. If you do not enter SAVE or END, you are terminated immediately. The EDIT work file is retained, if modifications have been made. If SAVE is issued and the attempt is unsuccessful, the edit session is terminated. However, the work file data set is retained if modifications were made, and you see a message.

See *z/OS TSO/E Messages* for more information about the messages in this part.

Recovering After a Terminal Line Disconnect

If your user profile specifies the RECOVER attribute, EDIT creates work files during your edit session, which can be used as input to recover any modification made to your data set in the event of a line disconnect or system failure.

Through the use of the RECOVER operand of the EDIT command, you are given the opportunity to recover the modifications made to your data set prior to the disconnect.

If your user profile specifies the NORECOVER attribute, the system attempts to copy your edited data set (with all changes) into a data set with an intermediate qualifier name of EDITSAVE. This data set can then be edited the next time you log on.

Appendix C. Accessibility

Accessible publications for this product are offered through IBM Documentation for z/OS (www.ibm.com/docs/en/zos).

If you experience difficulty with the accessibility of any z/OS documentation see How to Send Feedback to IBM to leave documentation feedback.

© Copyright IBM Corp. 1988, 2024

Notices

This information was developed for products and services that are offered in the USA or elsewhere.

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

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

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for IBM Documentation. Use of hyperlinks in other output formats of this information is at your own risk.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Site Counsel
2455 South Road

© Copyright IBM Corp. 1988, 2024

Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at ibm.com®/privacy and IBM's Online Privacy Statement at ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at ibm.com/software/info/product-privacy.

Policy for unsupported hardware

Various z/OS elements, such as DFSMSdfp, JES2, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those

products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: IBM Lifecycle Support for z/OS (www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and Trademark information (www.ibm.com/legal/copytrade.shtml).

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux®® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Index

Special Characters

A

B

C

D

E

F

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

W

Z

Product Number: 5655-ZOS

SA32-0971-60