Chapter Six

MANIPULATING FILES

Introduction

During the course of a user session in OpenVMS, it is often necessary for the user to manipulate files. There are several common DCL commands that every user should be able to use.

This lesson discusses the use of the COPY, PRINT, RENAME, DELETE, and PURGE commands.

Objectives

To manipulate files, a user should be able to:

- Make copies of files
- Display the contents of files
- Print files
- Rename files
- Remove files from disk

Making Copies of Files

Overview

You may want to move a copy of a file into another directory, to another disk, or even to another system.

One of the easiest and most common methods of moving files from one location to another is to use the COPY command.

This section discusses:

- The COPY command
- Modifying the COPY command
- Common errors in copying files
- The APPEND command

The COPY Command

To copy one or more existing files to a new file, use the COPY command. The COPY command can do the following:

- Copy an input file to an output file
- Concatenate two or more input files into a single output file
- Copy a group of input files to a group of output files

The COPY command has the following syntax:

```
$ COPY   input-filespec[,...]   output-filespec
```

- The input files remain unchanged.
- The output file contains the same text or data as the input file.

If you omit the input or output file specifications, you will be prompted for them as follows:

```
$ COPY
_From:   INPUT.TXT
_TO:     OUTPUT.FILE
```

To copy a file you must have read access to the file.

Sample COPY Operations and Commands

If you want to copy...	Use a command like this:
One file to another in the same directory	`$ COPY TEMPLATE.TXT -` `_$ APRIL.TXT`
One file to another in a different directory	`$ COPY MONTHLY.DAT -` `_$ [.REPORTS]MARCH.TXT`
Two or more input files into a single output file (concatenation)	`$ COPY DEPT1.DIS,-` `_$ DEPT2.DIS MASTER.DIS`
A group of input files to a group of output files in a different directory, using wildcards	`$ COPY *.TXT;* -` `_$ [8SM.NEWBOOK]*.*`
A file from a specified device and directory to your current default device and directory, keeping the same file name and file type	`$ COPY FAL$MISC_USER:[SOURCE]-` `_$ SAMPLE.TXT *.*`

A file from a specified node, device, and directory to your current default device and directory, keeping the same file name and file type	`$ COPY -`
	`_$ FAL$PRGLIB:[SOURCE]SAMPLE.TXT-`
	`_$ *.*`

Modifying the COPY Command

Qualifiers

You can add qualifiers to the COPY command to modify its action. Qualifiers can be combined to further tailor a copy operation.

If you want the system to...	Use this qualifier:
Request confirmation before each copy operation	**/CONFIRM**
Display the file specification of each file copied	**/LOG**
Set a specific protection on the output file	**/PROTECTION**

Interactive Exercise

Practice using the COPY command by copying the following files into the indicated directory in your user directory. Change the name of the destination file as indicated in the table.

File to be copied:	Destination Directory	Destination File Name
FAL$MAIN:TEMPLATE.FOR	[.CODE.FOR]	TEMPLATE_MAIN.FOR
FAL$SUB:TEMPLATE.FOR	[.CODE.FOR]	TEMPLATE_SUB.FOR
FAL$COM:TEMPLATE.COM	[.COM]	TEMPLATE.COM

Requesting Confirmation of the COPY Operation

Use the /CONFIRM qualifier to have the COPY command prompt the user for confirmation before each copy operation:

$ COPY/CONFIRM *filename.type filename.type*

You can use any combination of uppercase and lowercase letters for word responses, and they can be abbreviated.

- Abbreviations must be unique.
- Affirmative answers are YES, TRUE, and 1.
- Negative answers are NO, FALSE, 0, and the RETURN key.
- Enter QUIT or the CTRL/Z key combination to stop processing.
- When you respond by entering ALL, the command continues to process but no further prompts are given.
- If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

The following example shows a copy operation using the /CONFIRM qualifier

```
$ COPY/CONFIRM     *.DAT    [.DATA]*.*
COPY  WORK3:[USER1]HISTORY.DAT;1  to  WORK3:[USER1.DATA]
     HISTORY.DAT;1 ? [N]: N
COPY  WORK3:[USER1]INVENTORY.DAT;1  to  WORK3:[USER1.DATA]
     INVENTORY.DAT;1 ? [N]: Y
COPY  WORK3:[USER1]PROJECT.DAT;1  to  WORK3:[USER1.DATA]
     PROJECT.DAT;1 ? [N]: Y
$
```

Example 6-1 - Copy Operation Using /CONFIRM

Notes: Copy Operation Using /CONFIRM

1. Issue the command to copy all files having a file type of DAT to the DATA subdirectory. Specify the /CONFIRM qualifier.
2. This is the COPY confirmation message. Type N or press the RETURN key to specify that this file is not to be copied.
3. Type Y to request that this file be copied.

Displaying a Log of the COPY Operation

Use the /LOG qualifier to display the file specifications of each file copied:

$ COPY/LOG ***filename.type filename.type***

The following information will be displayed after the copy operation:

- The file specifications of the input and output files
- The number of blocks copied
- The total number of new files created

The following command copies all files having a file type of REP to the REPORTS subdirectory. A log of the operation is requested using the /LOG qualifier.

```
$ COPY/LOG     *.REP    [.REPORTS]*.*
%COPY-S-COPIED,  WORK3:[USER1]06DEC.REP;1  copied  to
     WORK3:[USER1.REPORTS]06DEC.REP;1  (7  blocks)
%COPY-S-COPIED,  WORK3:[USER1]13DEC.REP;1  copied  to
     WORK3:[USER1.REPORTS]13DEC.REP;1  (10  blocks)
%COPY-S-COPIED,  WORK3:[USER1]20DEC.REP;1  copied  to
     WORK3:[USER1.REPORTS]20DEC.REP;1  (9  blocks)
%COPY-S-NEWFILES,  3  files  created
$
```

Example 6-2 - Copy Operation Using /LOG

Notes: Copy Operation Using /LOG

1. This is the log message displayed when a file is copied
2. This is the summary log mesage displayed when all files matching the specification have been processed.

Setting a New Protection Code on a Copied File

Use the /PROTECTION qualifier to set a new UIC protection code on a copied file:

$ COPY/PROTECTION=(*prot-code*)
input-spec output-spec

- The default protection is that of the existing output file.
- If no output file exists, the current default protection applies.

Chapter Ten offers a more in-depth discussion on security and protection codes.

The following command copies the file MY.DAT to a new file called YOUR.DAT, specifying a new protection code.

```
$ COPY/PROTECTION=(W:RE,S:RWED,O:RWED,G:RWED) MY.DAT YOUR.DAT
```

Example 6-3 - Copy Operation Using /PROTECTION

Common Errors in Copying Files

Errors in copying files usually involve opening the input or the output file. Check the text of the error message to determine which kind of error has occurred.

The following is an example of an error in the input specification for the COPY command. In this example the input file specified was not found.

```
$ COPY SAMPLE.COM [.COMMANDS]*.*
%COPY-E-OPENIN, error opening WORK3:[STUDENT11]SAMPLE.COM; as
input -RMS-E-FNF, file not found
$
```

Example 6-4 - Error in Input File

The following is an example of an error in the output specification for the COPY command. The output directory has been secured with protection codes that are not satisfied by your privileges.

```
$ COPY TEST.LOG [STUDENT11]*.*
%COPY-E-OPENOUT, error opening WORK3:[STUDENT11]TEST.LOG;1 as
output -RMS-E-PRV, insufficient privilege or file protection
violation
%COPY-W-NOTCOPIED, WORK3:[USER1]TEST.LOG;1 not copied
$
```

Example 6-5 - Error in Output File

The APPEND Command

The APPEND command adds the contents of one or more input files to the end of the specified output file. The command is similar in syntax and function to the COPY command. The output file is expanded by the addition of the input file contents and the version number of the output file is not incremented.

The APPEND command has the syntax:

```
$ APPEND  input-file-spec[,...]
          output-file-spec
```

The ***input-file-spec*** parameter specifies the names of one or more files to be appended. Multiple input files are appended to the output file in the order in which they are specified. Multiple input files must be separated by a comma or a plus sign(+).

The ***output-file-spec*** parameter specifies the file to which the input files will be appended. You must specify at least one field in the output file specification. The APPEND command uses default device and directory information if none are supplied. If an asterisk is used as a wildcard in the output file specification, the corresponding fields of the first input file specification are used.

Displaying the Contents of Files

Overview

One way to display the contents of a file is to edit the file. An editor will display the file and allow you to scroll forward and backward through the document. When it is not necessary to move through a document, you may also use the TYPE command; this is a much quicker way to display the contents of a file at your terminal screen.

This section discusses the requirements, restrictions, and uses of the TYPE command.

The TYPE Command

Use the TYPE command to display the contents of a file at your terminal, using the following syntax:

```
$ TYPE file-specification
```

Displaying the Contents of a File

To display the contents of a file, you must have read access to the file. All files are created with default security protection.

Files that Should NOT be Displayed

If you attempt to display files that contain non-ASCII characters, such as page breaks, form feeds, or other control sequences that send instructions to the operating system, the operating system will try to carry out the instructions issued by the control characters. This may lock your terminal screen.

Executable images

- Easily identifiable by the .EXE file type

Data files

- Usually have .DAT file type
- Do not always contain non-ASCII characters (check the file)

Directory files

- Easily identifiable by the .DIR file type

Different Ways to Use the TYPE Command

Displaying Multiple Files

Use a **list of files** or **wildcards** to display the contents of more than one file.

List of files

- If more than one file is listed in the TYPE command, the files are displayed in the order specified.
- The following command will display the files in the order in which they are listed:

```
$ TYPE FILE03.TXT, FILE02.TXT, FILE01.TXT
```

Wildcards

- The files are displayed in alphabetical order.
- The command **`$ TYPE FILE*.TXT`** will display the files in the following order:

```
FILE01.TXT
FILE02.TXT
FILE03.TXT
```

Displaying One Screen at a Time

Use the TYPE/PAGE command to cause the contents of a file to be displayed one screen at a time:

`$ TYPE/PAGE` *`file-spec`*

When the screen is full, the system will display the following message at the bottom of the screen:

```
Press RETURN to continue
```

To abort the display, type any other character followed by the RETURN key.

If more than one file has been requested, you can cancel the display of the current file and continue with the next file by pressing the CTRL/Z key.

The default is `TYPE/NOPAGE`.

Displaying a File in a Different Location

If you have security access to a file and the directory that contains it, you can display the contents of the file in a different device, directory, or even on another system.

The following example demonstrates the use of the TYPE command to display the contents of a file on a different device and directory:

```
$  TYPE  FAL$DISK_TEST6:[SMITH]COSTS.TXT
```

The following example demonstrates the use of the TYPE command to display the contents of a file on a different system:

```
$  TYPE
PLUTO::FAL$DISK_KOPDATA:[SMITH]COSTS.TXT
```

Printing Files

Overview

Although there is movement in the direction of the paperless office, it is still important to be able to have a physical copy of some documents. The OpenVMS operating system allows users to print files by sending a copy of a file to a printer (called an output device).

Many users send copies of files to printers all the time. There are many different kinds of printers, some with special paper, some with forms. A computer system uses a system of queues, or lines of requests, to keep track of which jobs should be printing where and on what printer.

This section discusses:

- The PRINT command
- Modifying the action of the PRINT command
- Displaying information about print jobs
- Controlling print jobs
- Common errors in manipulating print jobs

PRINT Command

Printing Files

Use the PRINT command to send the contents of a file or files to an output device such as a printer. The printing of files is managed by the OpenVMS print manager. Print jobs are sent to a print queue that is managed by the print manager. The PRINT command dictates the file(s) to be printed, the printer queue that will manage the printing of the file(s) and the print characteristics.

The PRINT command has the following syntax:

```
$ PRINT filespec,[filespec...]
```

- You can supply a single file or a list of files (separated by commas) as the parameter to the PRINT command.
- The PRINT command uses a default type of `LIS` if another type is not specified.

Requirement for the PRINT Command

To print a file, you must have read access to the file. This means that the file must be on a disk and in a directory that you can read, and it must also be available for you to read.

Entry Number

When you issue the PRINT command, the OpenVMS operating system assigns an entry number to your print job. This number is used to identify the job and can be referenced by the user or system manager to control or manipulate the job in the print queue.

This number is displayed in an informational message when the PRINT command is issued. Entry numbers indicate the order in which jobs are queued.

In the following example, the system assigns entry number 776 to a print request.

```
$  PRINT  TEST.DAT
Job  TEST  (queue  SYS$PRINT,  entry  776)  started  on  LPA0
$
```

Example 6-6 - Using the PRINT command

Modifying the Action of the PRINT Command

Qualifiers for the PRINT Command

You can use a variety of qualifiers to modify the action of the PRINT command. Qualifiers can be combined to further customize a print operation. Some of the most common qualifiers are listed below.

If you want to...	Use this qualifier:
Specify a particular queue	**/QUEUE**
Specify the number of copies	**/COPIES**
Request notification when the print job completes	**/NOTIFY**

Specifying a Queue

There is often more than one printer connected to a system. A printer queue is created to communicate to each printer, and in some cases, multiple printer queues may be created to manage different types of jobs. For example, some printers may handle large jobs, some may handle small (quicker) jobs, and some may print out special forms. These queues will be given names by the system manager and can be referenced in the print request to send a particular job by using the /QUEUE qualifier:

```
$PRINT/QUEUE=LETTER_QUALITY  MONLTY_REPORT.TXT
```

If a queue is not specified, your file will be sent to SYS$PRINT, which is the default system queue.

Specifying the Number of Copies Wanted

Use the /COPIES=*n* qualifier to specify the number of copies you want.

- The variable *n* represents the number of copies and can be a number from 1 to 255. The default is 1.
- The following command prints two copies of the file MEMO.TXT:

  $ PRINT/COPIES=2 MEMO.TXT

The position of the /COPIES qualifier in the command determines its effect.

- When attached to the PRINT command, the /COPIES qualifier affects all files in the print job. For example, the following command prints two copies of each file designated:

  $ PRINT/COPIES=2 Q1.TXT, Q2.TXT
- When it is attached to a file specification, the /COPIES qualifier only affects that one file. The following command prints two copies of Q1.TXT and one copy of Q2.TXT:

  $ PRINT Q1.TXT/COPIES=2, Q2.TXT

Asking the System to Notify on Completion

Use the /NOTIFY qualifier to ask the system to broadcast a message at your terminal when a print job has completed. The following example illustrates the system responses as the LOGIN print job is entered into the print queue, starts printing, and completes printing.

```
$  PRINT/NOTIFY  LOGIN.COM
Job  LOGIN  (queue  SYS$PRINT,  entry  799)  started  on  LPS40$SCDTST
$
%LPS-I-JOBSTART,  Job  799  Start
$
Job  LOGIN  (queue  SYS$PRINT,  entry  799)  completed
$
```

Example 6-7 - Using the /NOTIFY Qualifier

Displaying Information About Print Jobs

You can monitor the progress of your print requests once they are in the printer queue; you can intervene to adjust your request as long as the print job has not begun to execute.

Use the SHOW ENTRY command or the SHOW QUEUE command to display the status of jobs in the print queue.

SHOW ENTRY Command

Use the SHOW ENTRY command to display information about your print jobs or about specific job entries:

```
$ SHOW ENTRY [entry-number,...] [job-name],...
```

If you do not specify an entry number or job name, only information about your queue entries will be displayed.

The display shows the current status and attributes of each entry, including entry number, job name, owner, job status, and queue name. The following example shows the output from the SHOW ENTRY command.

```
$ SHOW ENTRY
Entry   Jobname         Username        Blocks      Status
-------  ------------    --------------  ---------   ---------
  970   MONTHLY         USER1              25       Printing
        On busy printer queue LTA1
  973   EX_COM_PROC     USER1             170       Pending
        On Generic printer queue POST
$
```

Example 6-8 - Listing Queue Entries

The following example uses the SHOW ENTRY command to display information about a specific queue entry.

```
$ SHOW ENTRY 970
Entry  Jobname         Username        Blocks      Status
-------  ------------  --------------  ---------   ---------
  970  MONTHLY         USER1               25      Printing
       On busy printer queue LTA1
$
```

Example 6-9 - Displaying a Specific Queue Entry

Add the /FULL qualifier to display more detailed information. The following example uses the SHOW ENTRY/FULL command to display detailed information about all of your queue entries.

```
$ SHOW ENTRY/FULL
Entry     Jobname         Username       Blocks    Status
-------   -------------  -------------  ---------  ---------
  970     MONTHLY        USER1            25       Printing
     On busy printer queue LTA1
     Submitted 17-AUG-2001 12:35 /FORM=DEFAULT /PRIORITY=100
     File: _$1$DUA7:[USER1]MONTHLY.PS;1 (printing)
  973     EX_COM_PROC    USER1            170      Pending
     On Generic printere queue POST
     Submitted 17-AUG-2001 12:36 /FORM=DEFAULT /NOTIFY /PRIORITY=100
     File: _$1$DUA7:[USER1.BUILD]EX_COM_PROC.PS;3
$
```

Example 6-10 - Displaying Detailed Queue Information

SHOW QUEUE Command

The SHOW QUEUE command can display information about queues as well as about jobs in a queue.

$ SHOW QUEUE ***[queue-name]***

- If you do not specify a queue name, all queues will be displayed.
- Enter SHOW QUEUE/DEVICES to list only the printer queues.

The following example illustrates the output from the SHOW QUEUE command.

```
$ SHOW QUEUE
Printer queue LPS20$CRIER, on TIDY::CRIER, mounted form DEFAULT
Generic printer queue POST
Generic printer queue POSTO
Generic printer queue POST_20
  Entry   Jobname    Username      Blocks    Status
  -----   ---------- ------------ -------   ---------
    976   CHAP_ONE    USER1            265   Pending
    979   CHAP_TWO    USER1            287   Pending
     •
     •
     •
```

Example 6-11 - Displaying Queue Information

When the SHOW QUEUE command is issued without specification of a queue name or any qualifiers, all available queues will be displayed as well as the jobs submitted by this user.

Managing Print Jobs

You can take a variety of actions to manage a print job,such as changing its characteristics or removing the print request from the queue.

Changing the Characteristics of Your Print Job

Use the SET ENTRY command to change certain characteristics of your print job:

```
$  SET ENTRY   entry-number/qualifier
```

- The *entry-number* is your job number in the queue, displayed by the SHOW QUEUE or SHOW ENTRY command.
- Use *qualifier(s)* to specify the characteristic(s) you wish to modify.

Some commonly used qualifiers are listed below.

If you want to...	Use this SET ENTRY qualifier:
Delay processing until a specified time or date	/AFTER = time
Specify or change the number of copies	/COPIES= n
Hold the print request until a later time or date	/HOLD
Have the system notify you when the print job completes	/NOTIFY
Release a holding job for processing	/RELEASE

SET ENTRY will not affect a job that is currently executing.

The following example illustrates the use of the SHOW ENTRY command to change the processing time and number of copies for a print job. Notification of job completion is also requested.

```
$ PRINT SETHOST.LOG
Job SETHOST (queue SYS$PRINT, entry 749) pending
$ SHOW ENTRY/FULL
  Entry  Jobname   Username   Blocks  Status
  -----  -------   --------   ------  ------
    749  SETHOST   USER1          13   Pending
         On generic printer queue SYS$PRINT
         Submitted 14-AUG-2001 10:32 /FORM=DEFAULT /PRIORITY=100
         File: _$1$DUA1:[USER1]SETHOST.LOG;2
$ SET ENTRY/COPIES=2/NOTIFY/AFTER=11:00 749
$ SHOW ENTRY/FULL
  Entry  Jobname   Username   Blocks   Status
  -----  -------   --------   ------   ------
    749  SETHOST   USER1          26    Holding until 14-AUG 11:00
         On generic printer queue SYS$PRINT
         Submitted 14-AUG-2001 10:32 /FORM=DEFAULT /PRIORITY=100
         File: _$1$DUA1:[USER1]SETHOST.LOG;2 /COPIES=2
$
         •
         •
         •
Job SETHOST (queue SYS$PRINT, entry 749) completed
$
```

Example 6-12 - Changing Characteristics of a Print Job

In the following example, the SHOW ENTRY command is used to release a print job that is currently being held in the queue.

```
$ PRINT/HOLD MARCH_REPORT.TXT
Job MARCH_REPORT (queue SYS$PRINT, entry 112) holding
          •
          •
          •
$ SET ENTRY/RELEASE 112
$
```

Example 6-13 - Holding and Releasing a Print Job

Removing Your Print Job from the Queue

Use the DELETE/ENTRY command to remove your job from a print queue. This can be done while the job is waiting in the queue or while it is in progress.

$ DELETE/ENTRY=*(entry-number[,...])*

The *entry-number[,...]* parameter specifies the entry number (or a list of entry numbers) of the job(s) to be deleted.

- If you specify only one entry number, you can omit the parentheses.
- To find a job's entry number, enter the SHOW ENTRY or SHOW QUEUE command.

Requirements for Deleting a Print Job

The DELETE/ENTRY command requires one of the following:

- Operator (OPER) privilege
- Execute (E) access to the queue
- Delete (D) access to the specified job(s)

All users have delete access to any job(s) they have submitted. In addition, the following will have the privilege to manipulate user jobs: system managers, operators, and any other personnel with control of the system.

In the following example, the DELETE/ENTRY command is used to delete two print jobs, one that is pending and one that is holding. Note the system message that is displayed when each type of job is deleted.

```
$ SHOW ENTRY
  Entry  Jobname   Username   Blocks   Status
  -----  -------   --------   ------   ------
     25  CHAPTER8  USER1         976    Pending
         On generic printer queue SYS$PRINT
     27  SETHOST   USER1          26    Holding until 14-AUG 11:00
         On generic printer queue SYS$PRINT
$ DELETE/ENTRY=27
Job SETHOST (queue SYS$PRINT, entry 27) terminated with error
status
%JBC-F-JOBDELETE, job deleted before execution
$DELETE-I-DELETED, entry 27 aborting or deleted
$
```

Example 6-14 - Removing a Print Job from the Queue

Common Errors In Manipulating Print Jobs

When manipulating print jobs, common errors involve specifying nonexistent components such as files, queues, jobs, and entries.

In the following example, the user specifies a nonexistent file in the print request.

```
$ PRINT MARSH
%PRINT-E-OPENIN, error opening $1$DUA1:[STUDENT11]MARSH.LIS; as input
-RMS-E-FNF, file not found
%PRINT-F-CREJOB, error creating job
-JBC-E-EMPTYJOB, no file specified in job request
$
```

Example 6-15 - Nonexistent File Error

Here are some troubleshooting tips with regard to common printing errors:

- Check your typing: did you spell the file name correctly?
- Check your default directory: is the file you want in the current default directory?
- Check the file type of the file you want to print: if it is not LIS, you must specify the file type in your PRINT command.

In this next example, the user specifies a nonexistent queue in the print request.

```
$ PRINT/QUEUE=LPS40 LOGIN.COM
%PRINT-F-CREJOB, error creating job
-JBC-E-NOSUCHQUE, no such queue
$
```

Example 6-16 - Nonexistent Queue Error

To display the names of the print queues available on your system, issue the SHOW QUEUE/DEVICES command. Once you verify the queue you want to print to, you can issue the print request with confidence.

In this example, the print job entry number specified by the user is incorrect. As a result, the DELETE/ENTRY command fails.

```
$  DELETE/ENTRY=2
%DELETE-W-SEARCHFAIL, error deleting 2
-JBC-E-NOSUCHENT, no such job
$
```

Example 6-17 - Nonexistent Job Error

When specifying entry numbers verify that the entry number matches the print job you want to print or delete by issuing the SHOW ENTRY command. This command will display the jobs and their entry numbers. You can then issue the delete request with confidence.

Another common error is to specify the block size rather than the entry number. In the following example, note that the entry number is 735, and not 2 (which is the block size).

```
$ SHOW ENTRY
   Entry   Jobname    Username    Blocks    Status
   -----   -------    --------    ------    ------
     735    LOGIN     USER1            2   Pending
      On generic printer queue SYS$PRINT
$
```

Example 6-18 - Display Printer Queue Entries

In this example, the system message does NOT indicate an error. The user has no jobs in the printer queue. All print jobs may have already completed.

```
$ SHOW ENTRY
%JBC-E-NOSUCHENT, no such entry
$
```

Example 6-19 - Nonexistent Entry Error

Renaming Files

Overview

There are times when it is necessary to rename a file in order to save a specific version of a file or a snapshot of a file at a particular point in time, or to indicate differences in a routinely edited file (e.g. LOGIN.COM).

Renaming a file can be accomplished by using the RENAME command; with a single action, the file will be copied to a new file and the original file will be deleted.

This section discusses:

- The RENAME command
- Modifying the RENAME command
- Common errors when renaming a file

The RENAME Command

Use the RENAME command to move a file to another directory on the same disk or change its name, while deleting the file from its original location.

The syntax of the RENAME command is:

```
$  RENAME input-filespec  output-filespec
```

- The output-filespec may contain a different directory name than the input-filespec, but it must not specify a different disk.

- If you omit one or both of the file specifications, the RENAME command will prompt you for them:

```
$ RENAME
_From:   TEST.COM
_To:   PRACTICE.COM
```

To rename a file, you must have delete access to the file, as well as write access to the directory containing the new file.

Sample Rename Operations

If you want to...	Issue this command:
Change the highest version of the file FILE1.OBJ to FILE2.OBJ	`$ RENAME FILE1.OBJ FILE2`
Rename all versions of all files with the file type TXT to file type OLD	`$ RENAME *.TXT;* *.OLD;*`
Rename a file into another directory. (This actually moves a file from one directory to another.)	`$ RENAME TEST.COM [.COMMANDS]`

Using Version Numbers with the RENAME Command

All existing versions of a file must be renamed; otherwise, the disk will still contain copies of the older versions after the renaming operation has completed.

The RENAME command understands relative version numbers. The chart below illlustrates how the RENAME command translates zero, negative zero, and negative version numbers:

Highest version	=	;0
Second highest version	=	;-1
Tenth highest version	=	;-9
Lowest version number	=	;-0

In the following example, a relative version number is used in a RENAME command.

```
$ RENAME MONTHLY_REPORT.TXT;-1 .OLD
```

Modifying the RENAME Command

Qualifiers

You can add qualifiers to a RENAME command to modify its action. Qualifiers can be combined to further customize a RENAME operation.

If you want the system to...	Use this qualifier:
Request confirmation before each renaming operation.	**/CONFIRM**
Display the file specification of each file renamed.	**/LOG**

Requesting Confirmation of the Rename Operation

Add the /CONFIRM qualifier to the RENAME command to control whether a request is issued before each renaming operation.

```
$ RENAME/CONFIRM input-filespec output-filespec
```

When you request confirmation prior to renaming files you will be prompted for each file that matches the specification. Respond to the prompt in one of the following three ways:

- Type N or press the RETURN key to prevent renaming.
- Type Y or YES to have the file renamed.
- If you respond with ALL, you will rename all of the remaining files.

The following command renames all files named SAMPLE to DEMONSTRATION. The file types remain unchanged. The /CONFIRM qualifier is used to request confirmation of the operation.

```
$ RENAME/CONFIRM SAMPLE.* DEMONSTRATION.*
RENAME FAL$DISK:[STUDENT12]SAMPLE.FILE;1 to
  FAL$DISK:[STUDENT12]DEMONSTRATION.FILE; ? [N]: N
RENAME FAL$DISK:[STUDENT12]SAMPLE.INFO;1 to
  FAL$DISK:[STUDENT12]DEMONSTRATION.INFO; ? [N]: Y
$
```

Example 6-20 - Renaming Operation Using the /CONFIRM Qualifier

Displaying a Log of a Renaming Operation

Use the /LOG qualifier to cause the system to display the name of each file as it is renamed:

```
$RENAME/LOG input-filespec output-filespec
```

The following command renames all files having a file type of .DATA to the new file type .INFO. (Note that while the file types have changed, the file names have remained the same.) A log of the operation is requested.

```
$ RENAME/LOG *.DATA *.INFO
%RENAME-I-RENAMED, MICKEY:[STUDENT02]PRELIMINARY.DATA;1
  renamed to MICKEY:[STUDENT02]PRELIMINARY.INFO;1
%RENAME-I-RENAMED, MICKEY:[STUDENT02]SAMPLE.DATA;1 renamed
  to MICKEY:[STUDENT02]SAMPLE.INFO;1
%RENAME-I-RENAMED, MICKEY:[STUDENT02]TESTING.DATA;1
  renamed to MICKEY:[STUDENT02]TESTING.INFO;1
$
```

Example 6-21 - Renaming Operation Using the /LOG Qualifier

Common Errors In Renaming Files

Errors in renaming files are usually involve opening the input file or writing to the output file. Check the text of the error message to determine which kind of error has occurred.

The following is an example of an error in the input specification for the RENAME command. The user does not have delete access to the input file.

```
$ RENAME TEST.COM [.COMMANDS]*.*
%RENAME-E-OPENIN, error opening DONALD:[USER1]TEST1.COM;1 as input
-RMS-F-RMV, ACP remove function failed
-SYSTEM-F-NOPRIV, no privilege for attempted operation
$ DIRECTORY/PROTECTION TEST1.COM

Directory DONALD:[USER1]

TEST1.COM;1   (RWE,RWE,RE,)

Total of 1 file.
$
```

Example 6-22 - RENAME Command Input File Error

The following is an example of an error on the output specification for the RENAME command. The user does not have write access to the output directory.

```
$ RENAME TEST2.COM [.DATA]
%RENAME-E-OPENIN, error opening DONALD:[USER1]TEST2.COM;1 as input
-RMS-F-ENT, ACP enter function failed
-SYSTEM-F-NOPRIV, no privilege for attempted operation
$ DIRECTORY/PROTECTION DATA.DIR

Directory DONALD:[USER1]

DATA.DIR;1    (RE,RE,RE,RE)

Total of 1 file.
$
```

Example 6-23 - RENAME Command Output File Error

Removing Files from Disk

Overview

There are a couple of reasons for a user to remove files from disk areas.

Space is always a consideration. Since the OpenVMS operating system writes a new version of a file every time a user edits it, versions can pile up very quickly and consume disk space.

It is also important to limit the number of copies of each file so that the user can easily select his most recent version.

This section discusses:

- The DELETE command
- Modifying the DELETE command
- Common errors when deleting files
- The PURGE command
- Modifying the PURGE command
- Common errors when purging files
- Erasing the contents of deleted files

The DELETE Command

Use the DELETE command to remove files and free disk space.

```
$ DELETE file-name.file-type;version-number
```

To delete a file, you must have security access to the file and directory. This includes delete access to the file, as well as write access to the directory containing the file.

You must specify the version number of the file you want to delete. If you do not specify the file to be deleted, you will be prompted for it.

```
$ DELETE
_File:   TEST.COM;1
```

There is no UNDELETE command. Once you delete a file, you usually cannot get it back.

Specifying Version Numbers

If you want to delete...	Issue the command:
A specific version of a file.	**DELETE** *name.type;version* `$ DELETE INVENTORY.LIS;3`
The highest version of a file.	**DELETE** *name.type;* `$ DELETE INVENTORY.LIS;`
All versions of a file.	**DELETE** *name.type;** `$ DELETE INVENTORY..LIS;*`

Modifying the DELETE Command

You can add qualifiers to the DELETE command to modify its action. Qualifiers can be combined to further customize a delete operation.

If you want the system to...	Use this qualifier:
Request confirmation before each delete operation.	/CONFIRM
Display the file specification of each file deleted.	/LOG

Requesting Confirmation of a Delete Operation

The /CONFIRM qualifier is helpful when you use wildcards to specify the files to be deleted. This protects against accidental deletion of the wrong files.

You will be prompted for each file that matches the specification. Respond to the prompt in one of the following three ways:

- Type N, 0, or False or press the RETURN key to prevent deletion.
- Type Y, 1, or True to have the file deleted.
- If you respond with ALL, all of the remaining files will be deleted, and no further prompts will be displayed.

```
$ DELETE/CONFIRM *.INFO;
DELETE FAL$MISC:[STUDENT01]DEMONSTRATION.INFO;1 ? [N]:   Y
DELETE FAL$MISC:[STUDENT01]DEMONSTRATION.INFO;1 ? [N]:   Y
DELETE FAL$MISC:[STUDENT01]DEMONSTRATION.INFO;1 ? [N]:   Y
$
```

Example 6-24 - The DELETE/CONFIRM Command

Requesting a Log of a Delete Operation

When deleting information, it is advisable to closely monitor the process to ensure that the desired files are being deleted. Use the /LOG qualifier to cause the system to display the name of each file as it is being deleted.

Common Errors In Deleting Files

Two common causes for errors when deleting files are:

- Failure to specify a version number, and
- Lack of delete access to the file.

Check the text of the error message to determine the kind of error that has occurred.

No Version Number Specified

```
$ DELETE MEMO.TXT
%DELETE-E-DELVER, explicit version number or wild card required
$
```

Example 6-25 - No Version Number Specified

If you want to delete...	Issue the command:
The highest version	`$ DELETE MEMO.TXT;`
A specific version	`$ DELETE MEMO.TXT;3`
All versions	`$ DELETE MEMO.TXT;*`

Trying to Delete a Protected File

The following error occurred because the user does not have delete access to the specified file.

```
$ DELETE COMMANDS.DIR;1
%DELETE-W-FILNOTDEL, error deleting MICKEY:[USER1]COMMANDS.DIR;1
-RMS-R-PRV, insufficient privilege or file protection violation
$
```

Here are some troubleshooting tips with regard to common errors when deleting files:

- If you do not own the file:
  - arrange to be given sufficient privilege to delete the file,
  - request that the protection on the file be changed, or
  - ask the system manager or file owner to perform the deletion.

- If you are the owner of the file, change the protection on the file to give yourself D (Delete) access and reissue the DELETE command.

The PURGE Command

Whenever a file is modified in any way, the OpenVMS operating system will create a new version of it. Purging old versions of files helps to conserve disk space.

To purge a file, you must have delete access to the file, as well as write access to the directory containing the file. Use the PURGE command to remove all but the most recent (highest numbered) version of a file from the disk:

```
$    PURGE  [file-specification]
```

- Do not include a version number with the file specification.
- If you omit the file specification, all files in the current default directory will be purged.

Modifying the Purge Command

You can add qualifiers to the PURGE command to modify its action. Qualifiers can be combined to further tailor a purge operation.

If you want the system to...	Use this qualifier:
Request confirmation before each operation.	**/CONFIRM**
Display the file specification of each file deleted.	**/LOG**
Specify the number of versions of the file to be retained.	**/KEEP**

Requesting Confirmation of the Purge Operation

The /CONFIRM qualifier is helpful when you use wildcards to specify the files to be purged. This protects against accidentally deleting the wrong files.

$ PURGE/CONFIRM ***file-spec***

When you request confirmation of the purge operation, you will be prompted for each file that matches the specification. Respond to the prompt in one of the following three ways:

- To prevent the purging, press the Return key or type N, O, or F.
- To have the file purged, type Y, 1, or T.
- If you respond with ALL, all of the remaining files will be purged, and no further prompts will be displayed.

```
$ PURGE/CONFIRM *.DATA
DELETE DONALD:[USER1]SAMPLE.DATA;2 ? [N]:   N
DELETE DONALD:[USER1]SAMPLE.DATA;1 ? [N]:   Y
DELETE DONALD:[USER1]TRIAL.DATA;1 ? [N]:   Y
$
```

Example 6-26 - The Purge Operation Using the /CONFIRM Qualifier

Displaying a Log of the Purge Operation

Use the /LOG qualifier to cause the system to display the name of each file as it is deleted. The syntax for the /LOG qualifier is:

$ PURGE/LOG ***file-spec***

The following command purges all files having a file type of .INFO and requests a log of the operation.

```
$ PURGE/LOG *.INFO
%PURGE-I-FILPURG, DONALD:[USER1]SAMPLE.INFO;1 deleted (3 blocks)
%PURGE-I-FILPURG, DONALD:[USER1]TEST.INFO;5 deleted (3 blocks)
%PURGE-I-FILPURG, DONALD:[USER1]TEST.INFO;4 deleted (3 blocks)
%PURGE-I-FILPURG, DONALD:[USER1]TEST.INFO;3 deleted (3 blocks)
%PURGE-I-FILPURG, DONALD:[USER1]TEST.INFO;2 deleted (3 blocks)
%PURGE-I-FILPURG, DONALD:[USER1]TEST.INFO;1 deleted (3 blocks)
%PURGE-I-TOTAL, 6 files deleted  (18 blocks)
$
```

Example 6-27 - The Purge Operation Using the /CONFIRM Qualifier

Keeping More Than One Version in a File

Use the /KEEP qualifier to retain more than one version of a file.

$ PURGE/KEEP=*n* *[file-spec]*

- The parameter *n* indicates the number of versions you wish to keep.
- The default is to keep only one version of the file (the most recent or highest numbered version).

The following command purges a file called WISHLIST.TXT, keeping the two highest versions.

```
$ DIRECTORY WISHLIST.TXT

Directory DONALD:[USER1]

WISHLIST.TXT;4  WISHLIST.TXT;3    WISHLIST.TXT;2  WISHLIST.TXT;1

Total of 4 files.
$ PURGE/KEEP=2 WISHLIST.TXT
$ DIRECTORY WISHLIST.TXT

Directory DONALD:[USER1]

WISHLIST.TXT;4  WISHLIST.TXT;3

Total of 2 files.
$
```

Example 6-28 - The Purge Operation Using the /KEEP Qualifier

You can customize the PURGE command by combining qualifiers. In the following example, the /LOG and /KEEP qualifiers are combined.

```
$  DIRECTORY/COLUMNS=1  WISHLIST.TXT

Directory  DONALD:[USER1]

WISHLIST.TXT;18
     •
     •
     •
WISHLIST.TXT;3

Total of 16 files.
$  PURGE/LOG/KEEP=5  WISHLIST.TXT
%PURGE-I-FILPURG,  DONALD:[USER1]WISHLIST.TXT;13  deleted  (3  blocks)
     •
     •
     •
%PURGE-I-FILPURG,  DONALD:[USER1]WISHLIST.TXT;3  deleted  (3  blocks)
%PURGE-I-TOTAL, 11 files deleted   (33 blocks)
$
```

Example 6-29 - Combining Purge Operation Qualifiers

Common Errors in Purging Files

Four common causes for errors when purging files are:

- Specifying a version number
- Lack of delete access to the file
- Lack of write access to the directory containing the specified file
- File locked by another user

Check the text of the error message to determine the kind of error that has occurred.

The following error occurred because the user specified a version number with the PURGE command.

```
$ PURGE TEST.*;
%PURGE-I-PURGEVER, version numbers not permitted
$
```

Example 6-30 - Version Number Error in Purge Operation

Never include a semicolon (;) in the **PURGE** command.

The following error occurred because the user does not have delete access to the specified file.

```
$ DIRECTORY/PROTECTION ESTHER_COUNT.FOR

Directory DONALD:[9RB]

ESTHER_COUNT.FOR;3      (RE,RE,RE,)
ESTHER_COUNT.FOR;2      (RE,RE,RE,)
ESTHER_COUNT.FOR;1      (RE,RE,RE,)

Total of 3 files.
$ PURGE ESTHER_COUNT.FOR
%PURGE-W-FILNOTPUR, error deleting DONALD:[9RB]ESTHER_COUNT.FOR;2
-RMS-E-PRV, insufficient privilege or file protection violation
%PURGE-W-FILNOTPUR, error deleting DONALD:[9RB]ESTHER_COUNT.FOR;2
-RMS-E-PRV, insufficient privilege or file protection violation
$
```

Example 6-31 - No Delete Access When Attempting Purge Operation

The following error occurred because the user does not have write access to the directory containing the specified file. Delete access to the file itself is not sufficient.

```
$ DIRECTORY/PROTECTION CODE.DIR

Directory DONALD:[9RB]

CODE.DIR;1           (RE,RE,RE,)

Total of 1 file.
$ DIRECTORY/PROTECTION [.CODE]

Directory DONALD:[9RB.CODE]

ESTHER_COUNT.FOR;3        (RWED,RWED,RE,)
ESTHER_COUNT.FOR;2        (RWED,RWED,RE,)
ESTHER_COUNT.FOR;1        (RWED,RWED,RE,)

Total of 3 files.
$ PURGE [.CODE]
%PURGE-W-FILNOTPUR, error deleting DONALD:[9RB.CODE]ESTHER_COUNT.FOR;2
-RMS-E-PRV, insufficient privilege or file protection violation
%PURGE-W-FILNOTPUR, error deleting DONALD:[9RB.CODE]ESTHER_COUNT.FOR;2
-RMS-E-PRV, insufficient privilege or file protection violation
$
```

Example 6-32 - Error Purging a File in a Protected Directory

Erasing the Contents of Deleted Files

When a file is deleted, the area in which it was stored is returned to the system for future use. The information that was stored in that location still exists there until new information is written over it. A person with advanced knowledge of the operating system and its internal structures may be able to access the data in a file that has been deleted, but not overwritten.

To prevent unauthorized access to a deleted confidential file, use the /ERASE qualifier when issuing the DELETE or PURGE commands. This causes the storage location to be overwritten with a system specified pattern so that the confidential information no longer exists.

```
$ DELETE file-specification /ERASE
$ PURGE file-specification /ERASE
```

Summary

Concepts

Making Copies of Files

- To copy one or more existing files to a new file, use the COPY command.
- To copy a file, you must have read access to the file.
- You can use a variety of qualifiers to modify the action of the COPY command.
- Errors in copying files usually involve opening the input or the output file.
- To add files to the end of an existing file use the APPEND command.

Displaying the Contents of Files

- Use the TYPE command to display the contents of a file at your terminal.
- If you do not supply all parts of a file specification, the TYPE command assumes certain default values.
- To display the contents of a file, you must have read access to the file.
- Use lists of files or wildcards to display multiple files.
- Use the TYPE/PAGE command to display a file one screen at a time.

Printing Files

- To send the contents of a file to an output device such as a printer, use the PRINT command.
- To print a file, you must have read access to the file.
- When you issue the PRINT command, the OpenVMS operating system assigns an entry number to you print job.
- You can use a variety of qualifiers to modify the action of the PRINT command. Qualifiers can be combined to further customize a print operation.
- To display the status of jobs in a print queue, use the SHOW ENTRY command or the SHOW QUEUE command.
- To change certain characteristics of your print job, use the SET ENTRY command.
- To remove your job from a print queue, use the DELETE/ENTRY command. The job can be in progress or waiting in the queue.
- Common errors in manipulating print jobs often involve specifying a nonexistent file, queue, job, or entry number.

Renaming Files

- To change the name of a file or move a file to a different directory on the same disk, use the RENAME command.
- To rename a file, you must have delete access to the file and write access to the destination directory containing the renamed file.
- You can use qualifiers to modify the action of the RENAME command.
- Errors in renaming files usually involve opening the input file or writing the output file.

Removing Files from Disk

- To remove specified versions of a file from a disk, use the DELETE command. You must specify the version number in some manner.
- To delete a file, you must have delete access to the file and write access to the directory containing the file.
- You can add qualifiers to the DELETE command to modify its action.
- Qualifiers can be combined to further customize a delete operation.
- Errors in deleting files usually involve failure to specify a version number or lack of delete access to the file.
- To remove all but the most recent (highest numbered) version of a file from a disk, use the PURGE command. You must not specify a version number.
- To purge a file, you must have delete access to the file and write access to the directory containing the file.
- You can add qualifiers to the PURGE command to modify its action.
- Qualifiers can be combined to further customize a purge operation.
- Errors in purging files usually involve specification of a version number or lack of delete access to the file.
- For further protection, delete sensitive files with the /ERASE qualifier.

Commands

Making Copies of Files

COPY
: Copy one or more existing files to a new file.

APPEND
: Copy one or more files to the end of an existing file.

Displaying the Contents of Files

TYPE
: Display the contents of a file.

Printing Files

PRINT
: Send the contents of a file to an output device (a print queue).

SHOW ENTRY
: Display status information about jobs in a print queue.

SHOW QUEUE
: Display status information about print queues.

SET ENTRY
: Change the characteristics of your print job.

DELETE/ENTERY
: Remove your print job from a print queue.

Renaming Files

RENAME
: Change the name of a file or move the file to another directory on the same disk.

Removing Files from Disk

DELETE
: Remove files from a disk.

PURGE
: Remove previous versions of files from a disk.