3

Working with Files and Directories

Objectives

After completing this lesson, you should be able to:

- Determine your location in the directory structure
- View file content
- Copy and move files and directories
- Create and remove files and directories
- Search for files and directories

Lesson Agenda

- Determining your location in the directory structure
- Viewing file content
- Copying and moving files and directories
- Creating and removing files and directories
- Searching for files and directories

Viewing Directories

- A directory is a list of references to objects, that can include files, subdirectories, and symbolic links.
- Each reference consists of two components:
  - **A name:** The name that is used to identify and access the object.
  - **A number:** The number specifies the inode where the metadata about the object is stored.
- You can use various commands to display the current directory, view content of a directory, and change directories.

Determining the Current Directory

The `pwd` (print working directory) command identifies the full or absolute pathname of the current working directory.

```
$ pwd
/home/oracle
```

Note: The `pwd` command works the same in both Oracle Solaris and Oracle Linux, although the directory structures may be different.

Displaying the Directory Content

- The `ls` command displays the content of a directory. The syntax for the `ls` command is:

```
$ ls [options] … [filename]
```

- To display the contents of the `/home/oracle/lab` directory, enter the `ls` command.

```
$ ls
dante    dante_1 ... dir1 ... file1 ... fruit ... practice
```

- To display the contents of the `dir1` subdirectory, enter the `ls dir1` command.

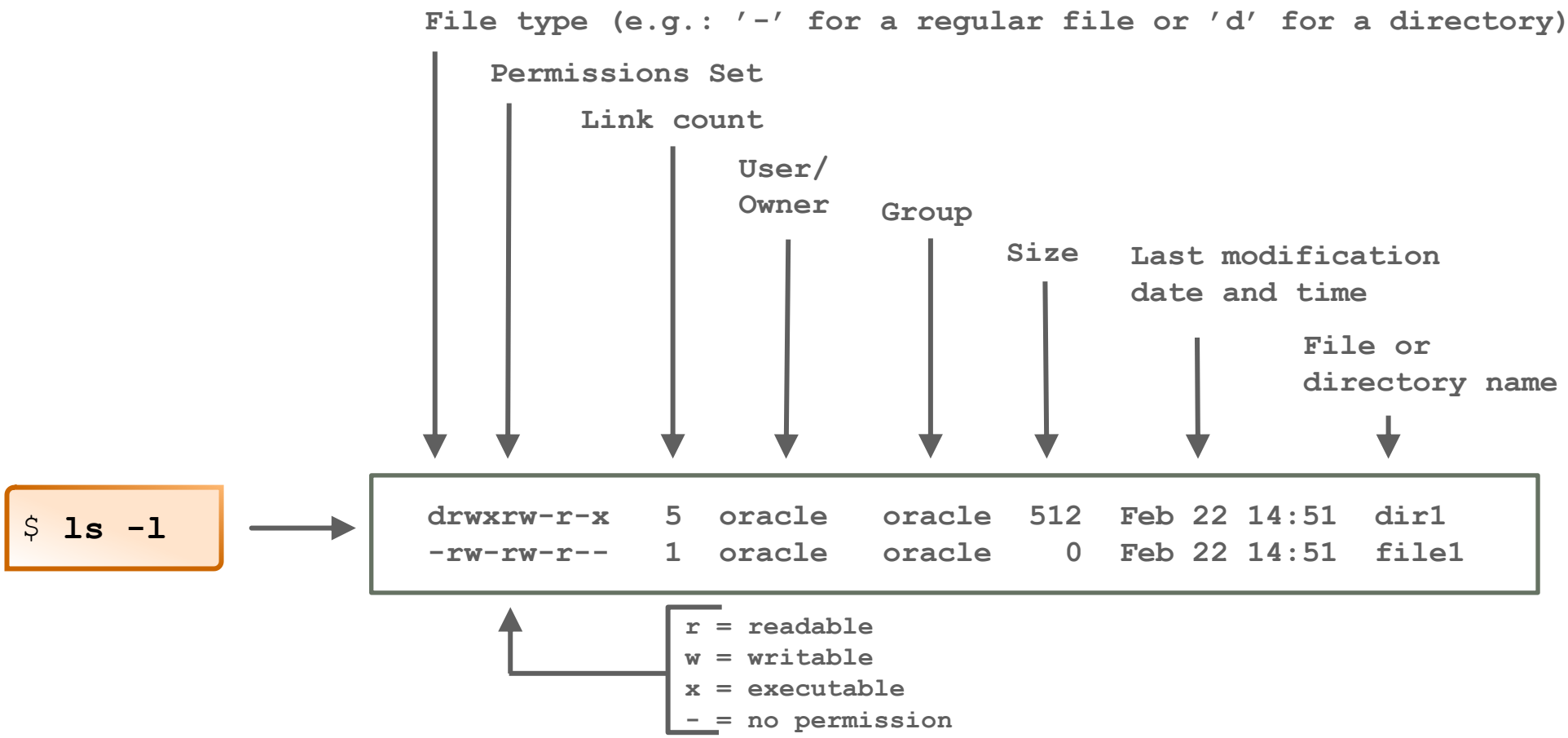
```
$ ls dir1
coffees  fruit   trees
```

- To display the contents of a directory, you can always use the full pathname to that directory.

```
$ ls /home/oracle/lab/dir2
beans    notes   recipes
```

Displaying the Directory Content with Options

- The `ls -l` command displays a `long` listing of file information.



Displaying the Directory Content with Options

- The `ls -a` command lists all files in a directory, including hidden files and hidden directories. Hidden files and directories are those that begin with a dot symbol or a period (.), sometimes called a dotfile.

```
$ ls -a
.               .gnome2_private       dante          file2
..              .gtkrc-1.2-gnome2     dante_1        file3
.ICEauthority   .metacity             dir1           file4
.Xauthority     .mozilla              dir2           fruit
.bash_history   .nautilus             dir3           fruit2
```

- The `ls -i` command lists the file's or directory's inode number and its corresponding name.

```
$ ls -i dante
12783  dante
```

Note: Your inode number may be different.

Displaying the Directory Content with Options

- The `ls -ld` command displays detailed information about a directory without showing its content.

```
$ ls -ld directory_name
```

- The `ls -R` command displays the content of a directory and all its subdirectories. This type of list is known as a recursive list.

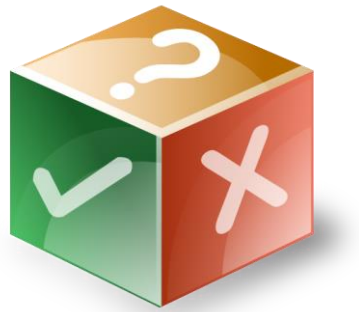
```
$ ls -R directory_name
```

- One of the differences between Oracle Solaris and Oracle Linux commands is that Solaris uses a lowercase "`-r`" for recursive operations and Linux sometimes uses an uppercase "`-R`".

Quiz

What is the function of the `ls -a` command?

a. Displays the content of a directory and all subdirectories of the directory
b. Displays only detailed information about the directory, not its content
c. Displays detailed information about the content of a directory, including hidden files
d. Displays all the files in a directory, including hidden files



Displaying File Types

- The `ls -F` command or the `file` command displays the type of file:

```
$ ls -F
dante   dir3/ file.2 file3 greetings
dante_1 dir4/ file.3 file4 myvars
$ file dir3
dir3: directory
```

- The following table shows the symbols or indicators used in the `ls -F` command output:

Indicator	File Type
*	Executable
/	Directory
=	Socket
@	Symbolic link
|	First In First Out (FIFO)

Displaying File Types

- The `file` command also helps to determine certain file types. The syntax for the `file` command is:

```
$ file [options] filename
```

- To view the file type for the `dante` file, enter the `file` command and specify the name of the file.

```
$ file dante
dante: ASCII English text
```

- To view the file type for the `/usr/bin/passwd` executable, enter the `file` command and specify the name of the file.

```
$ file /usr/bin/passwd
/usr/bin/passwd: setuid ELF 64-bit LSB shared object, ... dynamically linked ...
 stripped
```

- For more information about `file` command options, see the `file` man page.

Changing Directories

- When working within the directory hierarchy, you always have a current working directory.
- When you initially log in to the system, the current directory is set to your `home` directory.
- You can change your current working directory at any time by using the `cd` command:

```
$ cd directory
```

- When you use the `cd` command without options or arguments, the current working directory changes to your `home` directory.

Changing Directories by Using Relative or Absolute Pathname

- You can use either a relative or an absolute pathname to move around the directory hierarchy.
- A **relative** pathname lists the directories in the path relative to the current working directory.
- An **absolute** pathname lists all the directories in the path, starting with the root (/) directory.
- The following table describes the relative pathname:

Symbol	Pathname
.	Current or working directory
..	Parent directory, the directory directly above the current working directory

Home Directory

- The home directory of a regular user is where the user is placed after logging in.
- The user can create and store files in the home directory.
- Often the name of a user's home directory is the same as the user's login name.
  - If your user login name is `oracle`, your home directory on Oracle Linux would be `/home/oracle`, whereas your home directory on Oracle Solaris could be either `/export/home/oracle` or `/home/oracle`.

Returning to Your Home Directory

- You can return to your home directory by using one of the two methods:
  - Use the `cd` command without arguments:

```
$ cd
$ pwd
/home/oracle
```

  - Use the `cd` command with the absolute pathname to your home directory:

```
$ cd /export/home/oracle
or
$ cd ~
```

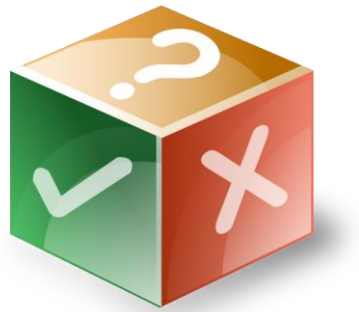
Note: “~” (tilde) is a shell metacharacter for one’s own home directory.

Quiz

When you use the `cd` command without options or arguments, the current working directory changes to your home directory.

a. True

b. False



Lesson Agenda

- Determining your location in the directory structure
- Viewing file content
- Copying and moving files and directories
- Creating and removing files and directories
- Searching for files and directories

Viewing File Contents

- There are several commands that display the content of a text file in read-only format.
- The file-viewing commands include the following:
  - `cat` (short for concatenate) can be used to concatenate several files into a single file. However, its most common usage is to display the contents of relatively small text files.
  - `more` is used to display (but not change) the contents of a text file one screen or page at a time.
  - `less` (sometimes called “GNU Less”) is much more robust than the legacy `more` command. `less` is used to display, navigate, and search (but not change) the text files’ contents, displaying one screen or page at a time.
  - `tail` displays lines of text at the end of the text file.
  - `head` displays lines of text at the beginning of the text file.
  - `wc` (word count) counts newlines, words, bytes, and characters in a text file.
  - `diff` displays the differences between two text files.

Viewing File Content: `cat` Command

- The `cat` command is most commonly used to display the content of text files.

```
$ cat [options] ... [filename]
```

- Use the `cat` command to display the contents of the `dante` text file.

```
$ cat dante
The Life and Times of Dante
by Dante Pocai
Mention “Alighieri” and few may know about whom you are talking.
Say “Dante,” instead, and the whole world knows whom you mean.
...(output truncated)
```

- For more information about `cat` command options, see the `cat` man page.

Note: Before you attempt to open a file by using the `cat` command, it is recommended that you first run the `file` command to determine the file type.

Viewing File Content: `more` Command

- The `more` command displays the content of a text file one screen at a time.

```
$ more [options] filename
```

- The `--More--(n%)` message appears at the bottom of each screen, where `n%` is the percentage of the file that has been displayed.
- When the entire file has been displayed, the shell prompt appears.
- For more information about `more` command options, see the `more` man page.

Viewing File Content: `more` Command

When the `--More--(n%)` prompt appears at the bottom of the screen, you can use the keys described in the table to scroll through the file.

Keyboard Command	Action
Space bar	Displays the next screen or page of the file you are viewing
Return / Enter	Displays the next line of the file you are viewing
b	Moves back one full screen
/pattern	Searches forward for a pattern (regular expression)
n	Finds the next occurrence of a pattern after you have used `/pattern`
N	Changes the direction of the search
h	Provides a description of all scrolling capabilities
q	Quits the `man` page and returns to the shell prompt

Viewing File Content: `less` Command

The GNU Project created `less` as a response to `more`, but with more features and actions. The table in the previous slide details not only `more` options but *some* of the `less` options also. For a more thorough list of features, entering `less -?` will yield greater details.

- The following is a snippet from the `less` man pages:

```
SYNOPSIS
     less -?
     less --help
     less -V
     less --version
     less [-[+]aABcCdeEfFgGiIJKLmMnNqQrRsSuUVwWX~]
          [-b space] [-h lines] [-j line] [-k keyfile]
          [-{oO} logfile] [-p pattern] [-P prompt] [-t tag]
          [-T tagsfile] [-x tab,...] [-y lines] [-[z] lines]
...(output truncated)
```

- For more information about `less` command options, see the `less` man page.

Viewing File Content: `head` Command

- The `head` command displays the first 10 lines of a file.

```
$ head [-n] filename
```

- You can change the number of lines displayed by using the `-n` option.
- To display the first **five** lines of the `/usr/dict/words` file in Solaris, enter the `head` command with the `-n` option set to `5`.

```
$ head -5 /usr/dict/words
10th
1st
2nd
3rd
4th
```

Note: In Linux, the directory file pathname is `/usr/share/dict/words`.

Viewing File Content: `tail` Command

- The `tail` command displays the last 10 lines of a file.

```
$ tail [options] filename
```

- You can change the number of lines displayed by using the `-n` or `+n` option.
  - The `-n` option displays `n` lines from the end of the file.
  - The `+n` option displays the file from line `n` to the end of the file.
- Possibly one of the more popular options is `-f`, which allows you to *follow* the data that is being appended to the end of the file.
- For more information about `tail` command options, see the `tail` man page.

Viewing File Content: `wc` Command

- The `wc` (word count) command displays the number of lines, words, and characters contained in a file.

```
$ wc [options] filename
```

- You can use the following options with the `wc` command.

Option	Description
`-l`	Line count
`-w`	Word count
`-c`	Byte count
`-m`	Character count

Viewing File Content Differences: `diff` Command

- The `diff` command displays the differences between two ASCII text files.

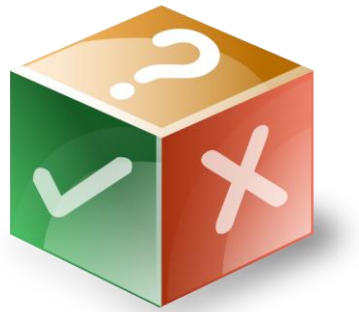
```
$ diff [options] filename1 filename2
```

- When the output of the `diff` command is displayed, the lines that are unique to *filename1* are identified by the < (less than) symbol, while lines that are unique to *filename2* are identified by the > (greater than) symbol. Lines that are identical in both files are not displayed.
- For more information about `diff` command options, see the `diff` man page.

Quiz

The default number of lines displayed by the `head` command is:

a. 5
b. 10
c. 15
d. 20



Lesson Agenda

- Determining your location in the directory structure
- Viewing file content
- Copying and moving files and directories
- Creating and removing files and directories
- Searching for files and directories

Copying Files and Directories

- The `cp` command copies single or multiple files and directories.

```
$ cp [options] source(s) target/destination, where source(s) can be multiple files(s) and target/destination can be a single file or directory.
```

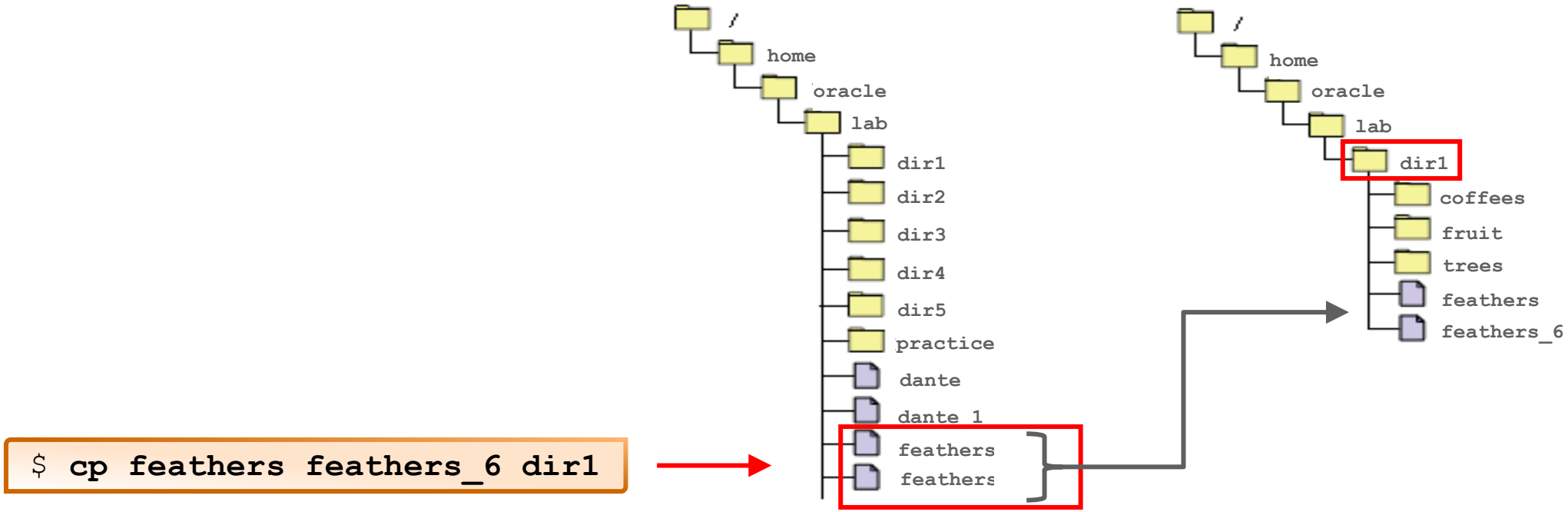
  - To copy a `filename` to a `newfilename` in the same directory, use the `cp` command with the name of the `source` file and the `target` file.

```
$ cp filename newfilename
```

- For more information about `cp` command options, see the `cp` man page.

Copying Multiple Files

- To copy multiple files to a different directory, use the `cp` command with multiple file names for the source and a single directory name for the target.
- The following graphic represents copying the `feathers` and `feathers_6` files to the `dir1` subdirectory.



Copying Files: `cp` Command Options

You can use the `cp` command with options and modify the functions of the command.

Option	Description
`-i`	Prevents you from accidentally overwriting existing files or directories
`-r` (Oracle Solaris) **`-R` (Oracle Linux)**	Recursive, includes the contents of a directory, and the contents of all subdirectories, when you copy a directory

Copying Files Recursively: `-r` or `-R` Option

- The `-r` or `-R` option recursively copies a directory.
- If the target directory does not exist, the `cp -r` command creates a new directory with that name.
- If the target directory exists, the `cp -r` command creates a new subdirectory with that name, below the destination directory.
- The sources option is one or more directory names. The target option is a single directory name.

Preventing Copy Overrides: `-i` Option

- The `-i` option prevents existing files from being overwritten with new files.
- When using the `-i` option, the system prompts for a **`y`**`es`/**`n`**`o` response from you before overwriting existing files.
- When copying the `feathers` file to the `feathers_6` file, the overwrite prompt appears:

```
$ cp -i feathers feathers_6
cp: overwrite ‘feathers_6‘? y
```

Moving and Renaming Files and Directories

- The `mv` command helps to move and rename files and directories within the directory hierarchy.

`$ mv [options] source(s) target/destination,` **where** `source(s)` **is the current filename or directory and** `target/destination` **is the new filename or directory.**

- The `mv` command does not affect the content of the files or directories being moved or renamed.
- The moved/renamed file maintains its original inode number.
- For more information about `mv` command options, see the `mv` man page.

Caution: `mv` is a **destructive** command if not used with the correct options.

Moving a File to Another Directory

Use the `mv` command to move the `brands` file from the `coffees` directory to the `/home/oracle` directory.

```
$ cd ~lab/dir1/coffees
$ pwd
/home/oracle/lab/dir1/coffees
$ ls
beans   brands  nuts
$ mv brands ~
$ ls
beans   nuts
$ cd
$ pwd
/home/oracle
$ ls -l brands
-rw-r--r-- 1 oracle oracle    0 Feb 6 2017 brands
```

Moving a Directory and Its Content

- You can also use the `mv` command to move a directory and its content to a different directory.
- Use the `mv` command to move the `practice` directory and its content to a new directory named `letters`.

```
$ cd ~/lab
$ pwd
/home/oracle/lab
$ ls -l practice
-rw-r--r-- 1 oracle  oracle              0 Feb 6      2017 mailbox
-rw-r--r-- 1 oracle  oracle       0 Feb 6     2017 project
$ mkdir letters
$ ls -l letters
total 0
$ mv practice letters
$ ls -l letters
drwxr-xr-x 2 oracle  oracle   512 Feb 6      14:11 practice
```

Renaming Files and Directories

- The `mv` command is also used for renaming existing files and directories.
- Use the `mv` command to rename the `dante` file `dantenew` in the current directory.

```
$ pwd
/home/oracle/lab
$ mv dante dantenew
$ ls
dante_1         dir2    feathers        file.3  file4   myvars
dantenew        dir3    feathers_6      file1   fruit   practice
```

Quiz

In your home directory, you created a directory called `newdir` as a placeholder for a variety of files. You notice that the directory now contains monthly reports. What would be the proper syntax to rename the `newdir` directory `monthly_reports`? (**Note:** You are not in your home directory.)

a. `mv ~/newdir ~/monthly_reports`

b. `mv newdir monthly_reports`

c. `mkdir ~/monthly_reports`

Lesson Agenda

- Determining your location in the directory structure
- Viewing file content
- Copying and moving files and directories
- Creating and removing files and directories
- Searching for files and directories

Creating Files

- The `touch` command creates a new empty file.

```
$ touch [options] filename
```

- You can create multiple files with the same command.
- You can also use the `touch` command to create `.hiddenfilenames`.
- If the file name or directory name already exists, the `touch` command updates the modification time and access time to the current date and time.
- You can use absolute or relative pathnames on the command line when creating new files.
- For more information about `touch` command options, see the `touch` man page.

Creating Directories

- The `mkdir` command creates new directories.

```
$ mkdir [options] directory_name
and
$ mkdir -p directory_name
```

- Include the `-p` option if the directory name includes a pathname, and the intermediate directories will also be created.
- You can use absolute or relative pathnames on the command line when creating new directories.
- For more information about `mkdir` command options, see the `mkdir` man page.

Creating Directories

- Create a new directory, named `Reports`, in the `/home/oracle` directory:

```
$ cd ~/lab
$ pwd
/home/oracle/lab
$ mkdir Reports
$ ls -ld Reports
drwxr-xr-x 2 oracle oracle  512 Feb 6 19:02 Reports
```

- Create a `Weekly` directory in the `Reports` directory:

```
$ mkdir Reports/Weekly
$ ls Reports
Weekly
```

Removing Files

- You can permanently remove files from the directory hierarchy by using the `rm` command.

```
$ rm [options] filename
```

- The `rm` command is a **destructive** command.
- The following table describes some of the options that you can use with the `rm` command when removing files and directories.

Option	Description
`-r` or **`-R`**	Recursive, includes the contents of a directory and the contents of all subdirectories when you remove a directory
`-i`	Prevents the accidental removal of existing files or directories by prompting for a '**y**es' or a '**n**o'
`-f`, `--force`	Forces the removal of existing files and directories (the opposite of –i) without prompting

- For more information about `rm` command options, see the `rm` man page.

Removing Files

- Remove the file named `projection` from the `letters` directory:

```
$ cd ~/lab/letters
$ ls
mailbox project projection research results
$ rm projection
$ ls
mailbox project research results
```

- Using `-i`, remove the contents of a directory:

```
$ cd ~/lab
$ rm -i file*
rm: remove regular file 'file1'? y
rm: remove regular file 'file2'? y
rm: remove regular file 'file3'? y
$ ls
```

Removing Directories

- You can use the `rm` command with the `-r` option to remove directories that contain files and subdirectories.

```
$ rm [options] directory
```

- Remove the `letters` directory and its content by using the `rm -r` command.

```
$ cd ~/lab
$ pwd
/home/oracle/lab
$ ls letters
mailbox results
$ rm -r letters
$ ls letters
ls:  cannot access letters: No such file or directory
```

Removing Empty Directories

- The `rmdir` command removes empty directories.

```
$ rmdir directory
```

- If a directory is not **empty**, the `rmdir` command displays the following error message:
  `rmdir: failed to remove "directory": Directory not empty`
- To remove a directory that you are currently working in, you must first change to its parent directory (`cd ..`).

Types of Links: Symbolic and Hard

- A **symbolic link** (sometimes called a symlink or a soft link) is a pointer that contains the full pathname to another file or directory.
- Symbolic links can link files and directories located across different file systems. The symlink makes the file or directory easier to access if it has a long pathname.
- Symlinks become “broken” if the linked to file is removed.
- A symbolic link file is identified by the letter `l` in the file-type field. To view symbolic link files, use the `ls -l` command.
- A **hard link** shares the inode of another file and increases the link count of the linked to file.
- Hard links must be on the same file system and hard links persist even if the other file is removed.
- To view hard link files, use the `ls -li` command and compare inode numbers and link counts.

Creating Symbolic Links

- You can use the `ln -s` command to create a symbolic link file.

```
$ ln -s source target
```

- You can use either relative or absolute pathnames when creating a symbolic link for a file.
- The filename for the symbolic link appears in the directory in which it was created.
- For more information about both symbolic and hard links using the `ln` command, see the `ln` man page.

Creating Symbolic Links

- Use the `ln -s` command to create a symbolic link file named `dante_link` to the `dante` file.

```
$ cd ~/lab
$ pwd
/home/oracle/lab
$ mv dante /var/tmp
$ ln -s /var/tmp/dante dante_link
```

- When using the `ls -F` command to display a list of files and directories, the symbolic links are identified with an @ symbol.

```
$ ls -F
Reports/      dir10/        feathers      file1*        fruit2
brands        dir2/         feathers_6    file2*        greetings
dante_1       dir3/         file.1*       file3*        myvars
dante_link@   dir4/         file.2*       file4*
```

Creating Hard Links

- You can use the `ln` command to create a hard link file.

```
$ ln source target
```

- You can use either relative or absolute pathnames when creating a hard link for a file.
- The filename for the hard link appears in the directory in which it was created.
- For more information about both symbolic and hard links using the `ln` command, see the `ln` man page.

Removing Both Symbolic and Hard Links

- You can use the `rm` command to remove both symbolic link files and hard link files, just as you would remove a standard file.
- Remove the `dante_link` symbolic link file by using the `rm` command.

```
$ ls -l dante_link
lrwxrwxrwx 1 oracle oracle ... dante_link -> /var/tmp/dante
$ rm dante_link
$ cat dante_link
cat: dante_link: No such file or directory
$ mv /var/tmp/dante dante
$ ls -l dante dante_link
ls: cannot access dante_link: No such file or directory
-rw-r--r-- 1 oracle oracle 1319 Feb 6 14:18 dante
```

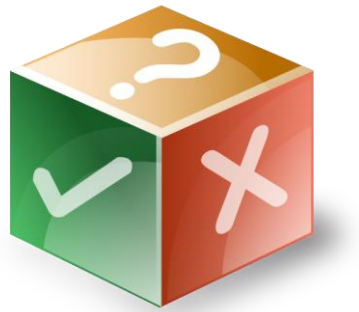
Quiz

Which command creates two new directories with the second directory as a subdirectory of the first?

a. `mkdir -i dir dir2`

b. `mkdir -p dir1/dir2`

c. `mkdir -r dir1/dir2`

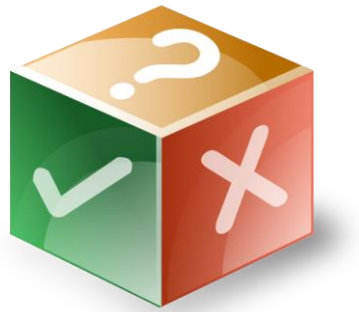


Quiz

The `rm -r` command removes nonempty directories.

a. True

b. False



Lesson Agenda

- Determining your location in the directory structure
- Viewing file content
- Copying and moving files and directories
- Creating and removing files and directories
- Searching for files and directories

Regular Expressions

- Usually, when searching in UNIX and Linux, a simple text string is all you need. However, when the pattern is more complex, you need *wild cards*.
- Regular Expressions (sometimes called regex or regexp) is a way to build those *wild card* patterns.
- Virtually, any userspace program that performs searches uses *regex_patterns*:
  - Example: `grep`, `more`, `less`, `vi`, `vim`, `emacs`, `sed`, `awk`, `tcl`, `ls`, `find`, `Microsoft Office`, and many more
- In addition, many languages, such as Perl, Java, PHP (Hypertext Preprocessor), Python, and Microsoft C and C++, use *regex_patterns*.

Regular Expressions Wild Cards

Wild cards (metacharacters)

- `*` Asterisk (glob) matches zero or more characters.
- `?` Question mark matches zero or a single character.
- `.` Period matches a single character.
- `^` Caret at the beginning of the line
- `$` Dollar at the end of the line
- `[ ]` Brackets (a *character class*). The characters inside the brackets match one character position.
- `' '` Single quotation marks (apostrophe) tell the shell to ignore any enclosed metacharacters.
- `" "` Double quotation marks enclose a space.
- `\` Backslash escapes the following metacharacter.

Creating `patterns` Using Regular Expressions

- The `ls` command lists all the files in the current directory.

```
$ ls
file1   file11    file2    file23   file3    file32    file4    file45
```

- If you want to list only `file2` and `file4`, you can build a *regex_pattern* `file[24]` (also called a *character class*):

```
$ ls file[24]
file2   file4
```

- Another example: If you want all files that end in 2 numerals, but the last numeral has to be a 3; one pattern could be `file[0-9][3]`:

```
$ ls file[0-9][3]
file23
```

Searching Files and Directories

- The `find` command searches for files that match the *filename_pattern*, (could be a *regex_pattern*) starting in the location specified by pathname.

```
$ find [pathname] -name filename_pattern
```

- The `find` command also searches for subdirectories recursively.
- When a file matches the *filename_pattern* specified in the `find` command, its full directory path is printed.
- One can search by `-name`, `-type` of file, `-perm` permissions, `-empty` or `-size`, `-group` name, `-user` username, and many others.
- For more information about `find` command options, see the `find` man page.

Searching Files and Directories: `find` Command

- Search for a file named `myscript.htm` in the current directory and any subdirectory.

```
$ find -name myscript.htm
```

- Search for any file named `mypage` beginning at the root directory (`/`) and all subdirectories from the root.

```
$ find / -name mypage
```

- To find all the files that start with `mess` located in the `/var` directory, add an asterisk (*) or glob.

```
$ find /var -name mess*
```

Using `expressions` with the `find` Command

- After the `find` command has found the files that you were searching for, often you may want to do something with those files. That's where `expression` comes in.

```
$ find [pathname] [expression]
```

- An expression could be the following:
  - Instead of getting only the full-directory-path for the files which were found, you may want to execute an `ls -l` to get a long listing of the found files.
  - Alternatively, you want to `cp` or `mv` the files.
  - The expression syntax can take many forms. You will explore two: the `-exec` and `-ok` that requires `y` or `Y` acknowledgment.

```
$ find [pathname] <what you're searching for> -exec <command> {} \;
```

- To get a long listing of the files found by `find` matching the *filename_pattern* `file*`:

```
$ find /opt –name file* -exec ls -l {} \;
```

Searching Files and Directories: Linux Also Has a `locate` Command

- To search using `locate`:

```
$ locate [options] regex_pattern
```

- Search for any file named `myscript.htm`:

```
$ locate myscript.htm
```

- The `locate` command works much like the `grep` command (covered in the following slides), to `locate` any files that contain the pattern "`mess*`".

```
$ locate mess*
```

- For more information about `locate` command options, see the `locate` man page.

Searching Within Files: `grep` Command

- The `grep` (global regular expression print) command allows you to search for a specified pattern in one or more files, printing any lines that contain the specified *pattern*.

```
$ grep [options] regex_pattern [files(s)]
```

- To search for the occurrence of “first” in a file called “Hello,” enter the following command:

```
$ grep first Hello
This is my first file in vim Editor
```

- For more information about `grep` command options, see the `grep` man page.

Searching Within Files: `grep` Command on Linux

- Sometimes it is useful to be able to highlight what was being searched for. The `grep` command on Linux has an option `--color`, which highlights the searched item in **red** in the output.
- To use the previous example, to search for the occurrence of “first” in a file called “Hello” highlighting what was found:

```
$ grep first --color Hello
This is my first file in vim Editor
```

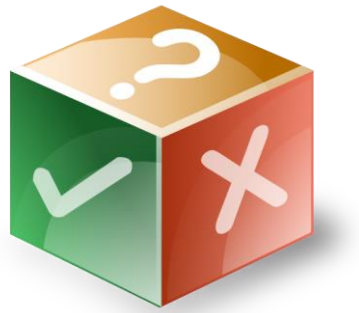
Quiz

Which of the following copy commands results in an error message?

a. `cp directory1 directory2`

b. `cp -r directory1 directory2`

c. `cp -ri directory1 directory2`



Summary

In this lesson, you should have learned how to:

- Determine your location in the directory structure
- View file content
- Copy and move files and directories
- Create and remove files and directories
- Search for files and directories

Practice 3: Overview

This practice covers the following topics:

- 3-1: Accessing files and directories
- 3-2: Using file and directory commands
- 3-3: Locating files and text