



Integrated Cloud Applications & Platform Services

UNIX and Linux Essentials

Student Guide

D76989GC20

Edition 2.0 | May 2018 | D103969

Learn more from Oracle University at education.oracle.com



Author

Al Flournoy

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

Technical Contributors

Craig McBride

Sebastien Colas

Corey Leong

Sreejith Mohan

Michael O'Reilly

Reviewers

Sebastien Colas

Craig McBride

Corey Leong

Michael O'Reilly

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Editors

Raj Kumar

Aju Kumar

Smita Kommini

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Graphic Designers

Prakash Dharmalingam

Kavya Bellur

Seema Bopaiyah

Publishers

Srividya Rameshkumar

Pavithran Adka

Syed Ali

Asief Baig

Michael Sebastian

Contents

1 Course Introduction

Lesson Objectives	1-2
Course Objectives	1-3
Course Schedule	1-4
Next Course to Take After the UNIX and Linux Essentials Course	1-5
Introductions	1-6
The Classroom Practice Environment	1-7
Your Desktop Environment	1-8
Summary	1-9
Practice 1: Overview	1-10

2 Introduction to the UNIX and Linux Environments

Objectives	2-2
Lesson Agenda	2-3
Multiuser, Multitasking, Time-Sharing Operating Systems	2-4
UNIX and Linux OS Structure	2-6
UNIX and Linux OS Structure: The Kernel	2-7
UNIX and Linux OS Structure: The File System	2-8
The File System	2-9
The root Directory “/”	2-10
File Types	2-11
UNIX and Linux OS Structure: Processes	2-12
UNIX and Linux OS Structure: The Shell	2-13
Shells, a User Command Interface to the Kernel	2-14
Quiz	2-15
User Accounts	2-16
Components of a User Account	2-17
Changing User Account Login Credentials	2-18
Changing User Account Login Credentials: Examples	2-19
Roles and Rights Profile	2-20
Changing Ownership of a Login Session to Another User/Role	2-21
User’s Home Directory	2-22
Quiz	2-23
UNIX Variants	2-24
Oracle Solaris	2-25

Linux Distributions	2-26
The Desktop Environment	2-27
Logging In to Oracle Linux Using the Desktop Login Screen	2-28
Oracle Linux Desktop	2-29
Logging In to Oracle Solaris Using the Desktop Login Screen	2-30
Oracle Solaris Desktop	2-31
Logging In Using the Command-line Option	2-32
Logging Out	2-33
Practice 2: Overview	2-34
Lesson Agenda	2-35
Executing Commands from the Command Line	2-36
Command-Line Syntax	2-37
Using UNIX Commands	2-38
Using Commands with Options	2-39
Using Commands with Options on Oracle Linux	2-40
Using Commands with Options in Oracle Solaris	2-41
Using Commands with Arguments	2-42
Using Commands with Options and Arguments	2-43
Using Multiple Commands on the Command Line	2-44
Quiz	2-45
Using the Man Pages	2-46
Displaying the Man Pages	2-47
Scrolling Through the Man Pages	2-48
Searching the Man Pages	2-49
Searching the Man Pages: By Section	2-50
Searching the Man Pages: By Keyword	2-51
Accessing Online Product Documentation	2-52
Quiz	2-53
Summary	2-54
Practice 2: Overview	2-55

3 Working with Files and Directories

Objectives	3-2
Lesson Agenda	3-3
Viewing Directories	3-4
Determining the Current Directory	3-5
Displaying the Directory Content	3-6
Displaying the Directory Content with Options	3-7
Quiz	3-10
Displaying File Types	3-11
Changing Directories	3-13

Changing Directories by Using Relative or Absolute Pathname	3-14
Home Directory	3-15
Returning to Your Home Directory	3-16
Quiz	3-17
Lesson Agenda	3-18
Viewing File Contents	3-19
Viewing File Content: cat Command	3-20
Viewing File Content: more Command	3-21
Viewing File Content: less Command	3-23
Viewing File Content: head Command	3-24
Viewing File Content: tail Command	3-25
Viewing File Content: wc Command	3-26
Viewing File Content Differences: diff Command	3-27
Quiz	3-28
Lesson Agenda	3-29
Copying Files and Directories	3-30
Copying Multiple Files	3-31
Copying Files: cp Command Options	3-32
Copying Files Recursively: -r or -R Option	3-33
Preventing Copy Overrides: -i Option	3-34
Moving and Renaming Files and Directories	3-35
Moving a File to Another Directory	3-36
Moving a Directory and Its Content	3-37
Renaming Files and Directories	3-38
Quiz	3-39
Lesson Agenda	3-40
Creating Files	3-41
Creating Directories	3-42
Removing Files	3-44
Removing Directories	3-46
Removing Empty Directories	3-47
Types of Links: Symbolic and Hard	3-48
Creating Symbolic Links	3-49
Creating Hard Links	3-51
Removing Both Symbolic and Hard Links	3-52
Quiz	3-53
Lesson Agenda	3-55
Regular Expressions	3-56
Regular Expressions Wild Cards	3-57
Creating patterns Using Regular Expressions	3-58
Searching Files and Directories	3-59

Searching Files and Directories: find Command	3-60
Using expressions with the find Command	3-61
Searching Files and Directories: Linux Also Has a locate Command	3-62
Searching Within Files: grep Command	3-63
Searching Within Files: grep Command on Linux	3-64
Quiz	3-65
Summary	3-66
Practice 3: Overview	3-67

4 Using the vim Editor

Objectives	4-2
Agenda	4-3
vim Editor: Introduction	4-4
Accessing the vim Editor	4-5
vim Editor: Overview	4-6
vim Editor Modes	4-7
Switching Between the Two Most Common Modes	4-8
Agenda	4-9
Viewing Files in Read-Only Mode	4-10
Moving the Cursor Within the vim Editor	4-11
Inserting and Appending Text	4-13
Text-Deletion Commands	4-14
Edit Commands	4-15
Quiz	4-16
Searching for and Substituting (Replacing) Text Within a File	4-17
Copy-and-Paste Commands	4-18
Save and Quit Commands	4-19
Session Customization	4-20
Session Customization Commands	4-23
Quiz	4-24
Summary	4-26
Practice 4: Overview	4-27

5 Using Features Within the Bash Shell

Objectives	5-2
Lesson Agenda	5-3
Shell Expansions	5-4
Brace Expansion	5-5
Tilde Expansion	5-6
Parameter Expansion	5-7
Command Substitution	5-8

Path Name Expansion and File Name Generation	5-9
Asterisk (*) Expansion Symbol	5-10
Question Mark (?) Expansion Symbol	5-11
Square Bracket ([]) Expansion Symbols	5-12
Quiz	5-13
Lesson Agenda	5-14
Shell Metacharacters	5-15
Command Communication Channels	5-16
File Descriptors	5-17
Redirection Metacharacters	5-18
Redirecting Standard Input (stdin)	5-19
Redirecting Standard Output (stdout)	5-20
Redirecting Standard Error (stderr)	5-21
Pipe Symbol	5-22
Using the Pipe Symbol	5-23
Redirecting Standard Output (stdout) by Using the tee Command	5-24
Quoting Symbols	5-25
Quiz	5-26
Practice 5: Overview	5-27
Lesson Agenda	5-28
Variables: Introduction	5-29
Displaying Local Shell Variables	5-30
Displaying Global Environment Shell Variables	5-31
Setting and Unsetting Shell Variables	5-32
Default Bash Shell Variables	5-33
Customizing bash Shell Variables: PS1	5-34
Customizing Shell Variables: PATH	5-35
Quiz	5-36
Lesson Agenda	5-37
Introducing Command History	5-38
Displaying Previously Executed Commands	5-39
Use the ! Command to Re-execute a Command Line from History	5-40
Use the !! Command to Repeat the Last Command	5-41
Searching the History Entries	5-42
Using the ! Command to Search for and Execute History Entries	5-43
Editing Commands on the Command Line	5-44
Invoking File Name Completion	5-45
File Name Completion with More Than a Single Solution	5-46
Lesson Agenda	5-47
User Initialization Files	5-48
Default User Initialization Files for the bash Shell	5-49

Configuring the `~/.bashrc` File 5-50
Rereading the `~/.bashrc` File 5-51
Quiz 5-52
Summary 5-53
Practice 5: Overview 5-54

6 Using Basic File Permissions

Objectives 6-2
Lesson Agenda 6-3
Securing Files and Directories 6-4
File and Directory Permissions (ACL) 6-5
Viewing Permission Categories 6-6
Permission Groups 6-7
Permission Sets 6-8
Interpreting File and Directory Permissions 6-9
Determining File or Directory Access Permissions 6-10
Interpreting the `ls -n` Command 6-11
Determining Permissions 6-12
Quiz 6-13
Lesson Agenda 6-14
Changing Ownership on Files or Directories 6-15
Changing Both username and group Ownership 6-16
Changing group Ownership 6-17
Quiz 6-18
Lesson Agenda 6-19
Changing Permissions 6-20
Changing Permissions: Symbolic Mode 6-21
Special File Permissions: Setuid, Setgid, and Stick Bit 6-23
Changing Permissions: Octal Mode 6-24
Quiz 6-28
Lesson Agenda 6-30
Umask: A bash Shell Built-in Command 6-31
Determining the umask Octal Value 6-32
Applying the umask Value 6-33
Changing the umask Value 6-34
Summary 6-35
Practice 6: Overview 6-36

7 Performing Basic Process Control

Objectives 7-2
Agenda 7-3

Process: Overview	7-4
Attributes of a Process	7-5
Process States	7-6
Process Subsystems	7-7
Agenda	7-8
Listing System Processes	7-9
Listing All Processes	7-10
Listing Process Trees in Oracle Linux	7-12
Listing Process Trees in Oracle Solaris	7-13
Quiz	7-14
Terminating a Process	7-15
Terminating a Process: kill Command	7-16
Common Signals and Their Uses	7-17
Terminating a Process: kill Command	7-18
Terminating a Process: pgrep and pkill Commands	7-19
Terminating a Process: pkill Command	7-20
Forcefully Terminating a Process: Signal 9 (SIGKILL)	7-21
Quiz	7-22
Summary	7-23
Practice 7: Overview	7-24

8 Using Advanced Shell Features in Shell Scripts

Objectives	8-2
Agenda	8-3
Jobs in the bash Shell	8-4
Job Control Commands	8-5
Running a Job in the Background	8-6
Bringing a Background Job to the Foreground	8-7
Quiz	8-8
The alias Command	8-9
Command Sequence	8-10
Predefined Aliases	8-11
User-Defined Aliases	8-12
Deactivating an Alias	8-13
Removing an Alias	8-14
Quiz	8-15
Shell Functions	8-16
Defining a Function	8-17
Invoking a Function	8-18
Shell Options	8-19
Activating the noclobber Shell Option	8-20

Deactivating the noclobber Shell Option	8-21
Quiz	8-22
Agenda	8-23
Shell Scripts	8-24
Determining the Shell to Interpret and Execute a Shell Script	8-25
Creating a Shell Script	8-26
Executing a Shell Script	8-27
Comments in a Shell Script	8-28
Positional Parameters in a Shell Script	8-29
Quiz	8-30
Checking the Exit Status	8-31
The test Command	8-32
Integer/Arithmetic test Comparison Operators	8-33
String test Comparison Operators	8-34
File test Comparison Operators	8-35
Using the test Command in an if Statement	8-36
How to Test/Debug a Shell Script	8-37
Conditional Expressions	8-38
The && Operator	8-39
The Operator	8-40
The if Statement	8-41
The if Statement Additional Syntax	8-42
The case Statement	8-43
Looping Constructs	8-44
The for Loop Statement	8-45
Shifting Positional Parameters in a Loop	8-46
The while (True) Loop Statement	8-47
The until (True) Loop Statement	8-48
Quiz	8-49
Summary	8-50
Practice 8: Overview	8-51

9 Archiving, Compressing, and Performing Remote File Transfers

Objectives	9-2
Agenda	9-3
File Archival: Introduction	9-4
The tar Command	9-5
The Common tar Command Options	9-6
Creating a tar Archive	9-7
Viewing the Table of Contents of a tar Archive	9-8
Extracting a tar Archive	9-9

Quiz	9-10
Agenda	9-11
File Compression	9-12
Compressing a File: gzip Command	9-13
Uncompressing a File: gunzip Command	9-14
Viewing a Compressed File: zcat Command	9-15
Viewing a Compressed File: gzcat Command	9-16
Archiving and Compressing Multiple Files: zip Command	9-17
Viewing and Uncompressing Archive Files: unzip Command	9-18
Compressing a File: bzip2 Command	9-19
Uncompressing a File: bunzip2 Command	9-20
Viewing a Compressed File: bzcat Command	9-21
Quiz	9-22
Practice 9: Overview	9-25
Agenda	9-26
Computer Networking: Introduction	9-27
Layout of a Basic Network	9-28
OpenSSH and Remote Network Connections	9-29
Using Secure Shell (ssh) for Remote Login	9-30
Copying Files and Directories Between a Local and Remote Host: scp Command	9-31
Reversing the Direction and Copying Files from a Remote Host to a Local Host	9-32
Copying Local Directories to and from a Remote Host	9-33
Quiz	9-34
File Transfer Protocol (FTP): Introduction	9-35
Using OpenSSH's sftp to Transfer Files Either Remotely or Locally	9-36
Before Using sftp to Transfer Files, One Needs to Know the File Type and the Destination to Which the File Is Being Transferred	9-37
Using dos2unix or unix2dos Commands to Convert the File's Format Based on the Destination to Which the File is Being Transferred	9-38
Using the sftp Commands	9-39
Transferring Files Using sftp	9-40
Transferring Multiple Files Using sftp	9-41
Quiz	9-43
Summary	9-45
Practice 9: Overview	9-46

10 Oracle Cloud Computing

Lesson Objectives	10-2
What Is Infrastructure as a Service (IaaS)?	10-3

What Is Oracle Private Cloud Appliance?	10-4
What Is Oracle OpenStack?	10-5
What Are Oracle Cloud Infrastructure Services?	10-6
Key Concepts and Terms Used in Oracle Cloud Infrastructure	10-8
Key Concepts and Terms	10-9
Oracle-Provided Images and Available Shapes	10-12
Key Concepts and Terms	10-13
Task Flow to Launch an Oracle Cloud Infrastructure Instance	10-14
Setting up a Virtual Cloud Network (VCN)	10-15
Working with VCN Subnets	10-16
Launching an Instance	10-17
Viewing Instance Details	10-18
Creating a Block Storage Volume	10-20
Attaching a Block Storage Volume to an Instance	10-21
Connecting a Block Storage Volume to an Instance's Guest OS	10-22
Oracle Cloud Computing Resources	10-25
Quiz	10-26
Summary	10-28

Course Introduction



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle University and Canadian Business School use only.

Lesson Objectives

After completing this lesson, you should be able to:

- Describe the course objectives
- Describe the course schedule
- Provide introductions
- Describe the classroom practice environment



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Welcome to the UNIX and Linux Essentials course. This course is designed for students who have not previously used UNIX or Linux and do not know the basic commands for navigating through the operating systems (OSes). To be proficient in Oracle Solaris or Oracle Linux, students need to have basic knowledge of the UNIX and Linux operating system structure, such as the file system hierarchy and shell concepts. In addition, students need to know how to build and execute basic UNIX and Linux commands from the command line in order to use the operating system. Students can apply the knowledge and skills from this course to both the Oracle Solaris and Oracle Linux OSes.

Course Objectives

After completing this course, you should be able to:

- Provide the basic UNIX skills and knowledge to successfully perform simple tasks in an Oracle Solaris 11 or an Oracle Linux 7 environment
 - Work with files and directories
 - Use the vi (vim) editor to create and modify files
 - Use commands within the bash shell
 - View and modify file and directory permissions
 - Manage processes
 - Use advanced shell features in shell scripts
 - Archive files and perform remote file transfer
 - Oracle Cloud Computing
- Provide some meaningful practices around the areas covered in this course



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Course Schedule

Session	Module
Day 1	Lesson 1: Course Introduction
	Lesson 2: Introduction to the UNIX & Linux Environments
	Lesson 3: Working with Files and Directories
Day 2	Lesson 4: Using the vim editor
	Lesson 5: Using Features Within the <code>bash</code> Shell
	Lesson 6: Using Basic File Permissions
Day 3	Lesson 7: Performing Basic Process Control
	Lesson 8: Using Advanced Shell Features in Shell Scripts
	Lesson 9: Archiving, Compressing and Performing Remote File Transfers
	Lesson 10: Oracle Cloud Computing



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The *UNIX and Linux Essentials* course consists of three days of lecture and practices. As part of each lesson, you will have the opportunity to apply what you have learned in a series of hands-on practices.

Note: The schedule may be adjusted by the instructor to meet the needs of the students.

Next Course to Take After the UNIX and Linux Essentials Course

- Interested in Programming courses available from Oracle?
 - Shell Programming
 - Perl Programming
 - Java Programming
- Interested in Operating System's courses available from Oracle?
 - Oracle Linux
 - Oracle Linux 5 & 6 System Administration
 - Oracle Linux 7 System Administration
 - Oracle Solaris 11 System Administration



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

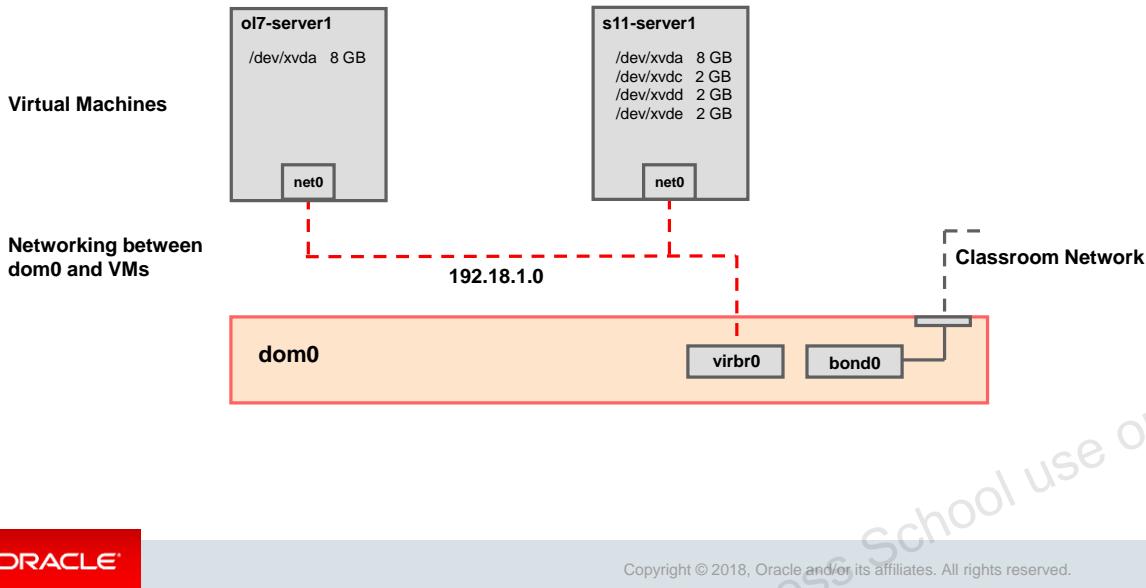
Introductions

- Name
- Company affiliation
- Title, function, and job responsibility
- Experience related to the topics presented in this course
- Reasons for enrolling in this course
- Expectations from this course



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

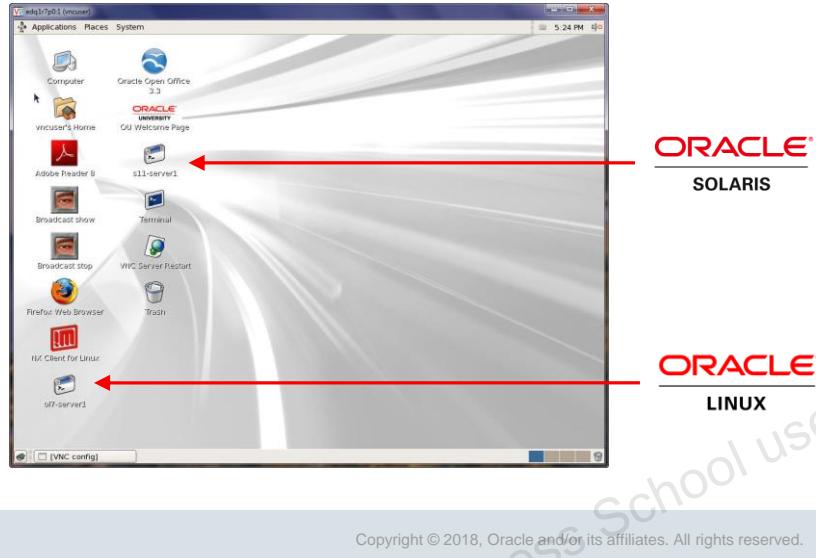
The Classroom Practice Environment



Your practice environment is based on the Oracle VM Server for x86 virtualization software installed on each student host machine known as `dom0`. The virtualization software extends the capabilities of your existing computer, so that you can run multiple operating systems inside individual virtual machines at the same time. As part of each lesson, you will be given the opportunity to practice in a lab environment that will be VMs running either Oracle Linux or Oracle Solaris.

Your Desktop Environment

This practice covers how to familiarize yourself with the lab environment.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Open your activity guide to Practices for Lesson 1, Course Introduction. Your instructor will walk you through the material and you will have a chance to familiarize yourself with the practice environment configuration and setup.

Summary

In this lesson, you should have learned how to:

- Describe the course objectives
- Describe the course schedule
- Provide introductions
- Describe the classroom practice environment



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Practice 1: Overview

This practice covers the following topics:

- 1-1: An introduction to your practice environment



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 1 in your Activity Guide.

Introduction to the UNIX and Linux Environments

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Describe the UNIX and Linux operating systems
- Execute commands from the command line



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

2 - 2

Lesson Agenda

- Describing the UNIX and Linux operating systems
- Executing commands from the command line



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Multiuser, Multitasking, Time-Sharing Operating Systems

- In 1965, Multics was being developed by MIT, Bell Labs (AT&T) and GE.
- In 1969, Bell Labs withdrew from the Multics Project and began work on its own UNIX OS.
- In 1977, programmers at the University of California, Berkeley made significant enhancements, including networking capability resulting in Berkeley Software Distribution (BSD UNIX).
- In 1983, Richard Stallman, formally of MIT's AI Lab, began the GNU Project to develop a free UNIX-like OS called GNU (GNU's Not UNIX).



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The first cut at a multiuser, multitasking OS was the Multics Project started in 1965 with contributions from Massachusetts Institute of Technology, Bell Labs (AT&T) and General Electric.

The UNIX OS was originally developed at AT&T Bell Laboratories beginning in 1969. It was created as a toolset by programmers for programmers. The early source code was made available to universities all over the country.

In 1977, the programmers at the University of California, Berkeley made significant modifications to the original source code from Bell Labs and called the resulting OS the Berkeley Software Distribution (BSD) UNIX. This version of the UNIX environment was sent to other programmers around the country, who added tools and code to further enhance BSD UNIX. Possibly the most important enhancement made to the OS by the programmers at Berkeley was adding networking capability. This enabled the OS to operate in a local area network (LAN).

In 1983, Richard Stallman, formerly of Massachusetts Institute of Technology's Artificial Intelligence (AI) Lab, began the GNU Project to develop a free UNIX-like OS called GNU (GNU's Not UNIX). The original kernel for the GNU OS was the Hurd kernel, still in development as recently as 2010.

Multiuser, Multitasking, Time-Sharing Operating Systems

- In 1987, AT&T UNIX, BSD UNIX, and other UNIX OSes were folded into System V Release 4 (SVR4) UNIX, that became an industry standard for the UNIX OS.
- In 1991, while the GNU OS was still in need of a kernel, Linus Torvalds, a Finnish Computer Science student, offered to create a kernel for the GNU OS.
 - In September, the Linux Kernel and the GNU userspace programs were released as the Linux OS.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

In 1987, AT&T UNIX, BSD UNIX, and other UNIX OSes were folded into what became System V release 4 (SVR4) UNIX. This was a new generation UNIX OS, which became an industry standard.

The new SVR4 UNIX became the basis for not only Sun and AT&T versions of the UNIX environment, but also IBM's AIX and Hewlett-Packard's HP-UX.

In 1991, while the GNU userspace programs were well on their way to being completed, the Hurd Kernel was still not working. A computer science student at the University of Helsinki, Linus Torvalds, offered to create a kernel for the Intel x86 architecture. In September 1991, Linus released the first version of the Linux kernel, which in conjunction with the GNU userspace programs became the Linux OS.

UNIX and Linux OS Structure

The UNIX and Linux OSes are structured around the following parts:

- Kernel
- File system
- Processes
- Shell

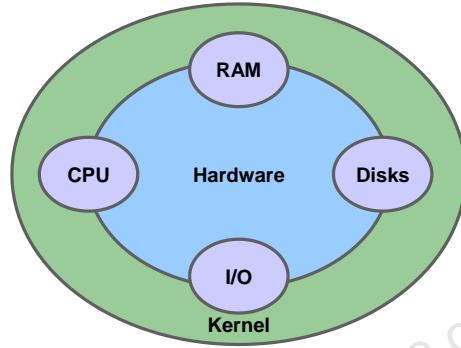


Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

UNIX and Linux OS Structure: The Kernel

The kernel is the core of the OS and manages all the physical resources of the hardware, including:

- Memory management
- Device and storage management
- File system I/O
- Process management or Central Processing Unit (CPU) functions



UNIX and Linux OS Structure: The File System

- All data in UNIX and Linux is organized into files.
- All files are organized into directories.
- These directories are organized into a file system.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The File System

- A file system is a logical collection of files on a partition, slice or disk.
- The UNIX and Linux file systems are a collection of files and directories that has the following properties:
 - It has a root directory (/) that contains other files and directories.
 - It has a boot directory (/boot) short for `bootstrap` which is used to boot up the operating system.
 - Each file or directory is identified by its name, the directory in that it resides, and a unique identifier, called an inode.
 - Each file system is self contained, in that there are no dependencies between one file system and another.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following file systems are available in Oracle Solaris:

- Disk-based file systems: HSFS, PCFS, UDFS, UFS, and ZFS
- Network-based file systems: NFS and SMB
- Virtual file systems: MNTFS, NAMEFS, OBJFS, SHAREFS, SPECFS, and SWAPFS
- Temporary file system (TMPFS)
- Loopback file system (LOFS)
- Process file system (PROCFS)

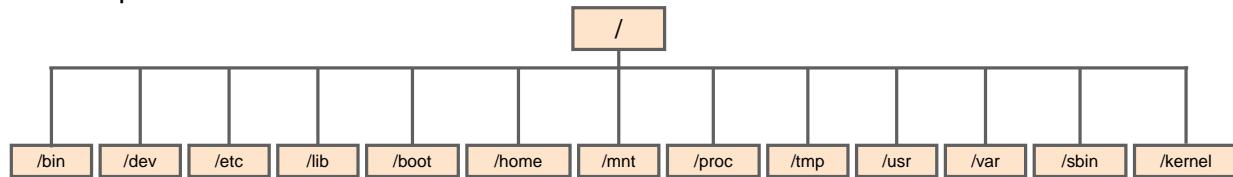
Note: UFS is a supported legacy file system, but it is not supported as a bootable root file system.

The following are the default file systems in Oracle Linux:

- UNIX file system having blocks, inodes and directories: ext2 (no longer used)
- ext2 file system enhanced with journaling capabilities: ext3
- ext3 further enhanced: ext4
- CDROM file system: isofs
- Oracle Cluster File System: ocfs, ocfs2
- Binary Tree File System: btrfs
- proc and sys file systems act as an interface to internal data structures in the Linux kernel
 - procfs and sysfs

The root Directory “/”

- UNIX uses a hierarchical file system structure, much like an upside-down tree, with the root (/) at the base of the file system and all other directories spreading from there.
- The directories have specific purposes and generally hold the same types of information.
- The following directories exist on both UNIX and Linux versions that are POSIX compliant:



Note: On first login, you are taken to your home directory.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The / in Oracle Solaris contains the following directories:

bin, dev, export, lib, net, platform, rpool, tmp, boot, devices, home, media, nfs4, proc, sbin, usr, cdrom, etc, kernel, mnt, opt, root, system, var

The / in Oracle Linux contains the following directories as defined by the LSB:

bin, etc, lib, lost+found, misc, net, proc, sbin, srv, tmp, var, boot, dev, home, lib64, media, mnt, opt, root, selinux, sys, usr

The Portable Operating System Interface (POSIX) is a family of standards specified by the Institute of Electrical and Electronics Engineers (IEEE) Computer Society for maintaining compatibility between operating systems.

File Types

- Everything in UNIX is considered to be a file, including physical devices, such as DVD-ROMs, USB devices, and external disks.
- In UNIX, there are three basic types of files:
 - Ordinary files (-)
 - Directories (d)
 - Special files
 - Symbolic links (l)
 - Character device (c)
 - Block device (b)
 - Socket (s)
 - Pipe (p)
 - Door (D)



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

In UNIX, there are three basic types of files:

- **Ordinary files:** An ordinary file is a file that contains data, text, or program instructions.
- **Directories:** Directories store both special and ordinary files.
- **Special files:** Some files that provide access to hardware such as hard drives, CD-ROM drives, modems, and Ethernet adapters are called special device files. Other special files are similar to aliases or shortcuts and enable you to access a single file using different names.

UNIX and Linux OS Structure: Processes

- Every program you run in the UNIX or Linux OSes creates a process.
 - When you log in and start the shell, you start a process.
 - When you run a command or when you open an application, you start a process.
- The system starts processes called daemons.
- Daemons are processes that run in the background and provide services.
 - For instance, the desktop login daemon provides a graphical prompt for logging in to the OS.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

UNIX and Linux OS Structure: The Shell

- The shell is primarily a command interpreter and serves as an interface between the user and the kernel.
- The shell accepts the commands that a user enters, interprets these commands, and passes them to the kernel, which executes the commands.
- The GNU Bourne Again Shell (`bash`) is the default shell in both Oracle Solaris and Oracle Linux.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Shells, a User Command Interface to the Kernel

The default shell in Oracle Linux and Oracle Solaris is:

- Bourne Again Shell (`bash`): Brian Fox (GNU Project 1989).

Other shells:

- Bourne (`sh`) - Stephen Bourne (Bell Labs 1977)
- C (`csh`) - Bill Joy (University of California, Berkeley 1978)
- TC (`tcsh`) - Ken Greer (Carnegie Mellon University 1981) an improved version of `csh`
- Korn (`ksh`) - David Korn (Bell Labs 1983)
- Z (`zsh`) - Paul Falstad (Princeton University 1990)
- Plus additional shells



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Available shells are:

- **Bourne Again Shell:** The GNU Project's `bash` shell is a Bourne-compatible shell that incorporates useful features from the Korn and C shells, such as command history, command-line editing, and aliasing.
- **Bourne:** The Bourne shell `sh` is one of the early UNIX system shells. The Bourne shell is the default shell for the root user/role. The root user is a special system account with unlimited access privileges for system administrators. The default Bourne shell prompt for a regular user is a dollar sign (\$). For the root user, the default shell prompt is a pound sign (#).
- **C:** The C shell `csh` has several features that the Bourne shell does not, such as command-line history, aliasing, and job control. The default C shell prompt for a regular user is the host name followed by a percent sign (`hostname%`). For the root user, the default shell prompt is the host name followed by a pound sign (`hostname#`).
- **TC:** The `tcsh` shell is a completely compatible version of the `csh` with additional enhancements.
- **Korn:** The Korn shell `ksh` is a superset of the Bourne shell with C shell-like enhancements and additional features, such as command history, command-line editing, aliasing, and job control. The default Korn shell prompt for a regular user is a dollar sign (\$). For the root user, the default shell prompt is the pound sign (#).
- **ZSH:** The Z shell is an extended version of the Bourne Shell `sh` with features from `bash`, `ksh`, and `tcsh` shells.

Note: Examples or codes used in this course assume the use of the Bash shell.

Quiz



Select the default shells in Oracle Linux and Oracle Solaris OSes:

- a. TC shell
- b. Z shell
- c. Korn shell
- d. C shell
- e. Bash shell
- f. Bourne shell



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

User Accounts

- A `user` account is a login account.
 - Regular users can log in and use the system, but cannot administer the system.
- A `group` is a collection of users who can share files and other system resources.
 - Users working on the same project could be formed into a group.
 - Each group must have a name, a group identification (GID) number, and a list of usernames that belong to the group. The GID number identifies the group internally to the system.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: In Oracle Solaris, you create and manage users, groups, and roles by using command-line tools only. There is currently no GUI tool for performing these tasks.

In Oracle Linux, you create and manage users, and groups, using command-line tools such as Oracle Solaris. Or by using `system-config-users` a GUI tool for performing those same tasks. Roles at this time are not supported in Linux.

Components of a User Account

The components of a user account are:

- Username
- Password
- User identification (UID) number
- Group identification (GID) number
 - Primary group
 - Secondary group
- Comment (GECOS)
- User's home directory
- User's login shell
 - Prompts for bash shell



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following are the seven colon-separated fields of a user account:

- **Username:** A unique name that a user enters to log in to a system. The username is also called the login name.
- **Password:** A combination of up to 256 letters, numbers, or special characters that a user enters with the login name to gain access to a system
- **User identification (UID) number:** A user account's unique numerical identification within the system
- **Group identification (GID) number:** A unique numerical identification of the group to which the user belongs. The two types of groups that a user can belong to are as follows:
 - **Primary group:** Specifies a group that the operating system assigns to files that are created by the user. Each user must belong to a primary group.
 - **Secondary groups:** Specifies one or more groups to which a user also belongs. Users can belong to up to 15 secondary groups.
- **Comment:** General Electric Comprehensive Operating System (GECOS) information that identifies the user
- **User's home directory:** A directory into which the user is placed after login. The home directory is the portion of a file system allocated to a user for storing private files.
- **User's login shell:** The user's work environment set up by the initialization files which are defined by the user's login shell
 - **Prompts for bash shell:** The default bash shell prompt for a regular user is a dollar sign (\$). For the root user/role, the default shell prompt is a pound sign (#).

Changing User Account Login Credentials

- As a matter of course a user's password expires on a regular basis.
- The `passwd` command allows you to update your password when it expires or on an as-needed basis.
- When changing a password, there can be new password requirements:
 - On the number of characters, numbers and symbols that can be used
 - On the minimum length of the password
 - And password reuse-ability



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Changing User Account Login Credentials: Examples

- An Oracle Linux example:

```
$ passwd
Changing password for <username>
Changing password <username>
(current) UNIX password:
New Password:
Retype new Password:
passwd: all authentication tokens updated successfully
```

- An Oracle Solaris example:

```
$ passwd
passwd: Changing password for <username>
New Password:
Re-enter new Password:
passwd: password successfully changed for <username>
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Roles and Rights Profile

- Unlike a user account, that is a *login* account, a `role` is not a login account.
- Role-based access control (RBAC) is a security feature for controlling user access to tasks that would normally be restricted to the `root` role.
- Each `role` is associated with a rights profile.
- Rights profiles are convenient collections of authorizations and other security attributes, commands with security attributes, and supplementary rights profiles.
- Once a user has signed on, if they have been granted the `root` rights profile, they can use the `su - root` to assume the `root` role.
- A user can assume only the roles that are assigned to the user's login account.

Note: Currently RBAC is only supported on **Oracle Solaris**. Although, Red Hat's Fedora Project has been testing RBAC as part of Security Enhanced Linux (SELinux) for several years.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Rights profiles offer a convenient way to group security attributes. These profiles, listed in `/etc/security/prof_attr`, can be assigned by the `root` role to any user account. The `root` role is assigned all privileges and all authorizations, so that it can perform all of the tasks, just as `root` can when `root` is a user.

Changing Ownership of a Login Session to Another User/Role

- You can use the `su` command to change the ownership of the current login session to another user or role.
- The `su` command is commonly used to switch to the `root` user (Linux) or assume the `root` role (Solaris).

```
$ su [-] [username]
```

- The `-` (dash) is used to instantiate the home directory and environment variables of the switched-to user.
- If the `[username]` attribute is omitted then the `root` username is the default.

```
$ su -
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

User's Home Directory

- When you first log in, you are taken to your home directory.
- The home directory is where you create and organize all your files and subdirectories.
- To go to your home directory, use the following commands:

```
$ cd  
or  
$ cd ~
```

- To go to the home directory of another user, use the following command:

```
$ cd ~username
```
- The ~ (tilde) is a pathname expansion of a user's home directory.

Note: Security settings within an organization might prevent users from accessing/viewing the home directory of another user.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The home directory is the portion of a file system that is allocated to a user for storing private files. A home directory can be located either on the user's local file system or on a remote file server. In either case, by convention, the home directory in Oracle Solaris could be created as /export/home/username or /home/username, and in Oracle Linux as /home/username.

To use a home directory from anywhere on the network, you should always refer to the home directory as \$HOME, not as /export/home/username or /home/username, because these are machine specific. In addition, any symbolic links that are created in a user's home directory should use relative paths so that the links are valid no matter where the home directory is mounted.

Quiz



A role in Oracle Solaris is a login account just like a user account.

- a. True
- b. False



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

UNIX Variants

Several variants evolved out of UNIX. The following are some of the popular UNIX variants:

- BSD (Berkeley Software Distribution)
- SunOS (BSD - Predecessor of Solaris)
- Oracle Solaris (AT&T's Bell Labs - System V)
- AIX (IBM - System V)
- Mac OS X (Apple - BSD)
- HP-UX (Hewlett Packard - System V)
- UNIXWare (Novell - System V)



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: This course covers only Oracle Solaris and Oracle Linux.

Oracle Solaris

- Sun's original version of the UNIX Operating System was known as SunOS, based on BSD UNIX version 4.2.
- At that time, AT&T's version of the UNIX environment was known as System V.
- AT&T, BSD, Sun and others created SVR4.
- SVR4 with multiprocessor capability became Oracle Solaris.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Linux Distributions

- Linux is packaged and distributed as a Linux distribution (distro) for desktop and server.
- There are over 600 Linux distros (see DistroWatch.com).
- Linux is a UNIX-like OS assembled and developed by Richard Stallman's GNU Project supplying the userspace programs and the Linux kernel from Linus Torvalds under the open source software development and distribution model.
- Some popular Linux distributions include:
 - Red Hat Enterprise Linux (RHEL) including Fedora, CentOS, and others
 - Oracle Linux (OL) based on RHEL with Oracle's Unbreakable Enterprise Kernel (UEK)
 - Debian GNU/Linux including Ubuntu, Linux Mint, and others
 - Slackware
 - openSUSE (SUSE)
 - Gentoo



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle Linux, formerly known as Oracle Enterprise Linux, is a Linux distribution based on Red Hat Enterprise Linux, repackaged and distributed by Oracle Corporation and available under the GNU General Public License (GPL) since late 2006.

Oracle Linux can be freely downloaded through Oracle's E-delivery Cloud service, and can be deployed and distributed free of cost.

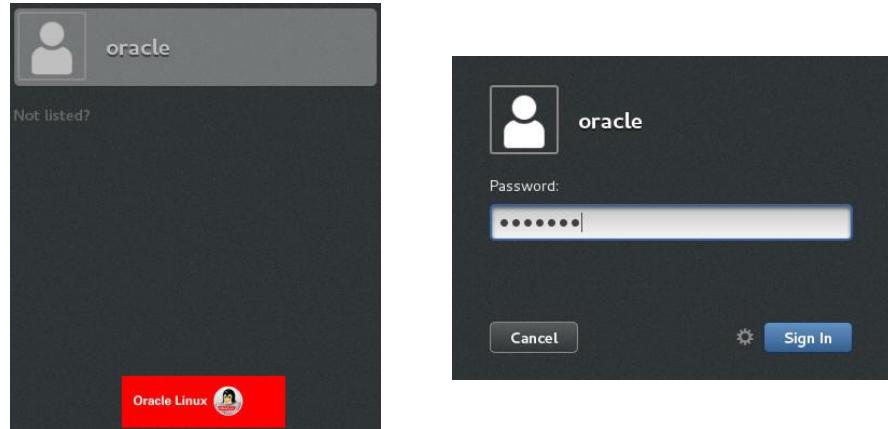
The Desktop Environment

- The desktop environment is a graphical user interface (GUI) that allows you to perform a range of activities, such as:
 - Securing and selecting sessions
 - Adding and deleting workspaces
 - Changing backgrounds
 - Managing files
- The look and feel of Oracle Solaris and Oracle Linux desktop environments are displayed in the subsequent slides.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Logging In to Oracle Linux Using the Desktop Login Screen



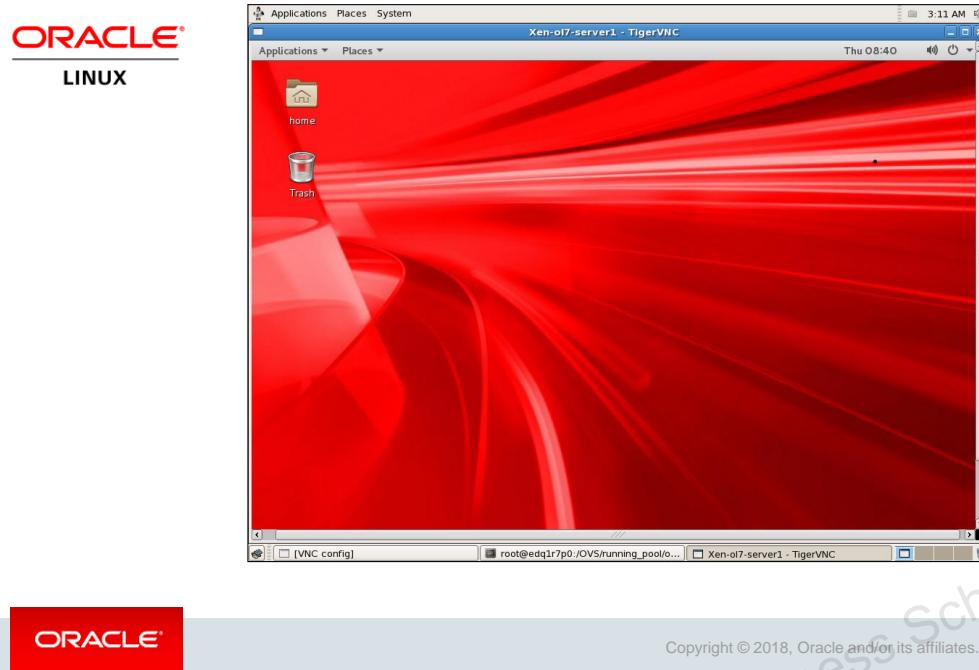
ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To log in to an Oracle Linux desktop session, perform the following steps:

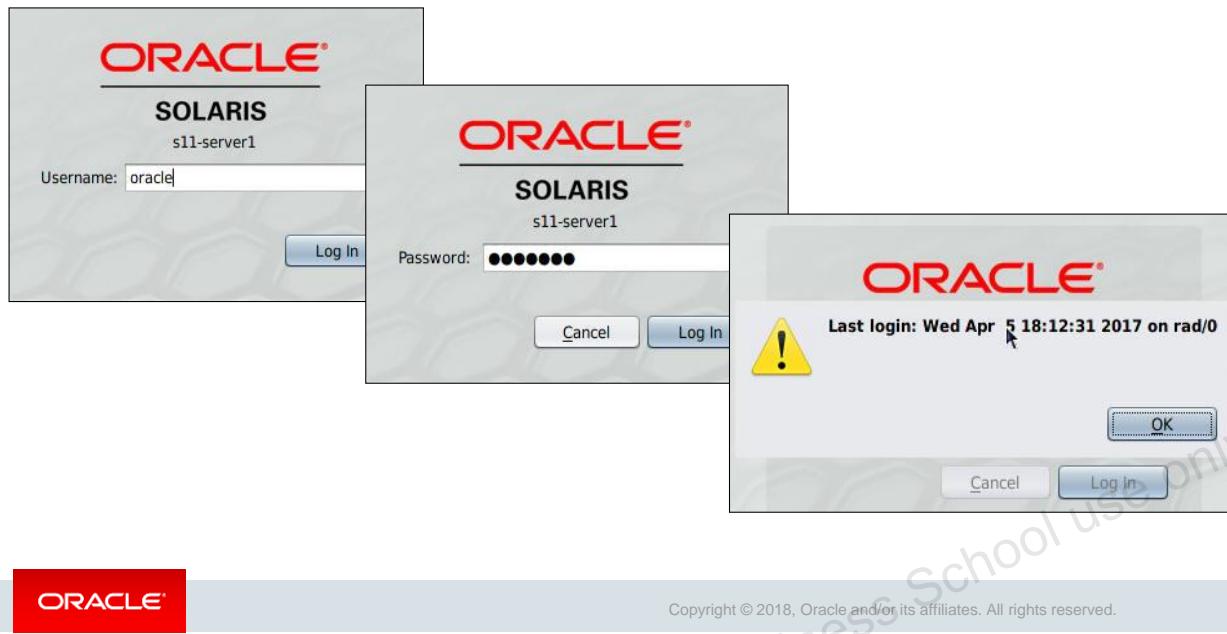
1. On the Login screen, click the `oracle` username.
2. Next, enter your password.
3. Press Return/Enter or click **Sign In**. Retry, if the login attempt fails.

Oracle Linux Desktop



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Logging In to Oracle Solaris Using the Desktop Login Screen



All users must follow a login process so that the system can recognize and authenticate the user. The desktop Login window as displayed in the slide enables you to log in to the system and use the desktop.

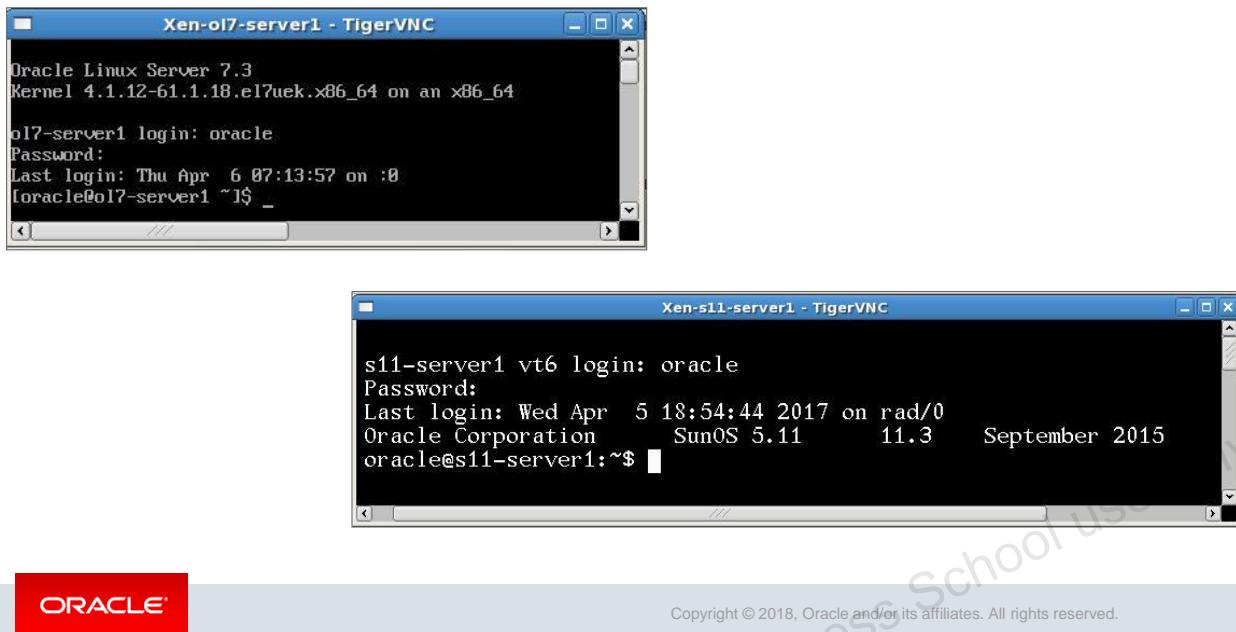
To log in to a desktop session, perform the following steps:

1. On the Login screen, enter your username.
2. Press Return/Enter or click **Log In**.
3. Next, enter your password.
4. Press Return/Enter or click **Log In**. Retry, if the login attempt fails.
5. If you receive a confirmation popup window, click **Ok**.

Oracle Solaris Desktop



Logging In Using the Command-line Option



You also have the option of logging in using the command-line mode.

From **Oracle Linux** Desktop sign in/login screens pressing the Ctrl + Alt + F6 keys switches to the command-line mode. At the login prompt, you can log in with your user credentials. Pressing either the Ctrl + Alt + F1 or Alt + Right Arrow keys reverts to the desktop window.

From **Oracle Solaris** Desktop sign in/login screens pressing the Ctrl + Alt + F6 switches to the command-line mode. Pressing the Ctrl + Alt + F7 keys reverts to the desktop window.

Note: In some of the virtual machine (VM) environments, these key strokes may be different.

Logging Out

- Depending on the interface, you use different commands or steps to log out.
- To log out of the graphical user interface:
 1. In Solaris on the desktop window, click **System**. In Linux, click the down arrow on the far right-hand side of the top task bar.
 2. Next, in Solaris click **Log Out <username>**, or in Linux just click the **<username>** and then click **Log Out**. A logout confirmation window appears.
 3. Click **OK** or **Log Out** or just press **Enter** to log out.
- To log out of the command-line interface:
Type `exit`. This causes your shell to exit, or stop running.

Note: Some shells, depending on your configuration, also log you out if you type the EOF (End-Of-File) character, **Ctrl + D**.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

When you are done using the system, you should log out. This prevents other people from accidentally or intentionally getting access to your files.

Both Oracle Solaris and Oracle Linux follow the same procedure to log out.

Practice 2: Overview

This practice covers only the highlighted topics:

- 2-1: Logging in to the system and changing your user login password
- 2-3: Using the man pages
- 2-2: Displaying system information using the command line



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 2 in your Activity Guide.

Lesson Agenda

- Describing the UNIX and Linux operating system
- Executing commands from the command line

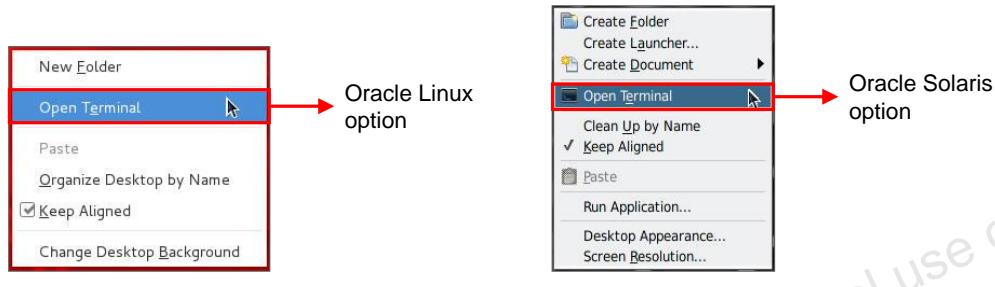


ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Executing Commands from the Command Line

- You can use system commands on the command line to instruct the system to perform specific tasks.
- The commands are entered into a terminal window.
- Right-clicking the Desktop causes a popup window to appear. Selecting *Open Terminal* will launch the terminal window.



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To open a terminal window, perform the following steps:

1. Move the cursor to an open space on the desktop.
2. Right-click the desktop. A pop-up menu appears.
3. Click **Open Terminal** in Oracle Solaris or Oracle Linux. A terminal window appears with a shell prompt.

Note: UNIX and Linux commands are case-sensitive.

Command-Line Syntax

- The command syntax is the structure and order of the command components: command name, options, and arguments.
- Command-line commands can exist with or without options or arguments.
- You can change the behavior of commands by using a combination of options and arguments.

Item	Description
Command	Specifies what the system does (an executable)
Option	Specifies how the command runs (a modifier). Options start with a dash (-) symbol.
Argument	Specifies what is affected (a file, a directory, or text)



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide describes the components of a command.

Note: Some Linux commands can sometimes use a double hyphen (--) for their command-line switches.

Using UNIX Commands

- The philosophy of UNIX commands is to create small, efficient programs that do a specific task and then group (pipe) them together to accomplish larger activities.
- There are over 4500 UNIX commands that can be run at the command line.
- To display the operating system information:

```
$ uname  
SunOS
```

- To display the date and time:

```
$ date  
Fri Mar 24 16:01:47 UTC 2017
```

- To clear the terminal window:

```
$ clear
```

- While there are many command-line commands, one regularly only uses just a couple of dozens of those commands. Most of which will be covered in this course.

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Some examples of basic commands are `uname`, `date`, and `clear`. The `uname` command provides information about the system. By default, when you use the `uname` command, the name of the current operating system appears.

Using Commands with Options

- Adding options to a command alters the information displayed.
- You can use more than one option with a command.
- You can also list multiple options separately or they can be combined after a dash (-).
- Use of a dash (-) preceding an option is command specific. Also, options are command specific.
- To get help with any specific command, one can enter <command> --help as an option.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

```
$ uname --help
```

Usage: uname [OPTION]...

Print certain system information. With no OPTION, same as -s.

-a, --all	print all information, in the following order, except omit -p and -i if unknown:
-s, --kernel-name	print the kernel name
-n, --nodename	print the network node hostname
-r, --kernel-release	print the kernel release
-v, --kernel-version	print the kernel version
-m, --machine	print the machine hardware name
-p, --processor	print the processor type or "unknown"
-i, --hardware-platform	print the hardware platform or "unknown"
-o, --operating-system	print the operating system
--help	display this help and exit
--version	output version information and exit

GNU coreutils online help: <<http://www.gnu.org/software/coreutils/>>
For complete documentation, run: info coreutils 'uname invocation'

Using Commands with Options on Oracle Linux

- The given example shows the `uname` command with two options:
 - The `-i` option displays the name of the hardware platform.
 - The `-n` option prints the host name of the local system.

```
$ uname -i  
x86_64  
$ uname -n  
ol7-server1
```

- The following example shows the `uname` command with two combined options.

```
$ uname -rs  
Linux 4.1.12-61.1.18.el7uek.x86_64
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: Some Linux commands can sometimes use a double hyphen (--) for their command-line switches.

Using Commands with Options in Oracle Solaris

- The given example shows the `uname` command with two options:
 - The `-i` option displays the name of the hardware platform.
 - The `-n` option prints the host name of the local system.

```
$ uname -i  
i86PC  
$ uname -n  
s11-server1
```

- The following example shows the `uname` command with two combined options:

```
$ uname -rs  
SunOS 5.11
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following example shows the `uname` command with `-s` and `-r` options. The `-s` option shows the name of the operating system. The `-r` option shows the operating system release level.

```
$ uname -s -r  
SunOS 5.11  
$
```

The following example shows the `uname` command with the `-a` option, which displays the current system information.

```
$ uname -a  
SunOS s11-server1 5.11 11.3 i86PC i386 i86PC
```

Using Commands with Arguments

- Arguments enable you to additionally define the output from a command.
- The following example shows the `cal` command with two arguments, `12` and `2015`.
 - The first argument, `12`, specifies the month.
 - The second argument, `2015`, specifies the year.

```
$ cal 12 2015
December 2015
S   M   Tu   W   Th   F   S
      1   2   3   4   5
 6   7   8   9   10  11  12
13  14  15  16  17  18  19
20  21  22  23  24  25  26
27  28  29  30  31
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Using Commands with Options and Arguments

Using the `ls` command without an option, with an option, with an argument, and with an option and argument together.

```
$ ls
dante      dir3          file.2          file3      greetings
dante_1    dir4          file.3          file4      myvars
$ ls -l
total 94
-rw-r--r-- 1 oracle      1319 Feb 6 09:25 dante
-rw-r--r-- 1 Oracle      368  Feb 6 09:25 dante_1
...(output truncated)
$ ls dante
dante
$ ls -l dante
-rw-r--r-- 1 oracle      1319 Feb 6 09:25 dante
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The `ls` command lists the files in a directory. The `-l` option provides additional information about the files. The filename argument specifies the file to be viewed.

Using Multiple Commands on the Command Line

- You can enter multiple commands on a single command line by using a semicolon (;) to separate each command.

```
$ command [options] [argument]; command [options] [argument]
```

- The shell recognizes the semicolon (;) as a command separator.
- The following example shows two commands separated by a semicolon.

```
$ date; uname  
Fri Mar 24 16:01:47 UTC 2017  
SunOS
```

- The shell executes each command from left to right when you press Enter/Return.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following example shows three commands separated by semicolons. The `cal` command has two arguments, followed by the `date` command, and the `uname` command with two options.

```
$ cal 12 2015; date; uname -rs  
December 2015  
S M Tu W Th F S  
1 2 3 4 5  
6 7 8 9 10 11 12  
13 14 15 16 17 18 19  
20 21 22 23 24 25 26  
27 28 29 30 31
```

```
Fri Mar 24 16:01:47 UTC 2017  
SunOS 5.11
```

Quiz



Given the `uname -a` command, which type of UNIX command component does `-a` represent?

- a. Option
- b. Parameter
- c. Argument



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Using the Man Pages

- The online technical reference manual (`man`) pages provide detailed syntax, descriptions, and usage of the commands.
 - You can use the `man command` to display the man page entry that explains a given command.
- ```
$ man command
$ man [options] command
$ man [options] filename
```
- For more information about the `man` command options, see the `man` page.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Displaying the Man Pages

To display the man pages for the `uname` command using the `man` command.

```
$ man uname
Reformatting page. Please Wait... done
User Commands uname(1)
NAME
 uname - print name of current system
SYNOPSIS
 uname [-aimnprsvX]
 uname [-S system_name]
DESCRIPTION
The uname utility prints information about the current system
on the standard output. When options are specified, symbols
representing one or more system characteristics will be written
to the standard output. If no options are specified, uname
prints the current operating system's name. The options print
selected information returned by uname(2), sysinfo(2), or both.
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Scrolling Through the Man Pages

The following table lists the keyboard commands for scrolling through the man pages.

| Keyboard Command      | Action                                                       |
|-----------------------|--------------------------------------------------------------|
| <b>h</b>              | Provides a description (help) of all scrolling capabilities  |
| <b>Space bar</b>      | Displays the next screen of a man page                       |
| <b>Return / Enter</b> | Displays the next line of a man page                         |
| <b>b</b>              | Moves back one full screen                                   |
| <b>/pattern</b>       | Searches forward for the <i>pattern</i> (regular expression) |
| <b>n</b>              | Finds the next occurrence of the <i>pattern</i>              |
| <b>N</b>              | Changes the direction of the search                          |
| <b>q</b>              | Quits the man command and returns to the shell prompt        |

**Note:** The `man` command uses the `less` command to scroll through the man pages. More about the `less` command in the lesson titled “Working with Files and Directories.”



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Searching the Man Pages

There are two ways to search for information in the man pages:

- Searching by section number
- Searching by keyword

| Section | Solaris Description       | Section | Linux Description         |
|---------|---------------------------|---------|---------------------------|
| 1       | User Commands             | 1       | User Commands             |
| 2       | System Calls              | 2       | System Calls              |
| 3       | Library Functions         | 3       | Library Functions         |
| 7       | Devices and Special files | 4       | Devices and Special files |
| 4       | File Formats              | 5       | File Formats              |
| 5       | Miscellaneous             | 7       | Miscellaneous             |
| 1M      | System Administration     | 8       | System Administration     |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The section numbers and meanings are different between Oracle Solaris and Oracle Linux.

## Searching the Man Pages: By Section

- The online man page entries are organized into sections based on the type or usage of the command or file.
  - Section 1 contains user commands.
  - Section 4 in Oracle Solaris and Section 5 in Oracle Linux contains information about various file formats.
- In Oracle Linux to look up a specific section of the man page, use the `man` command with the section number, and the command or file name.

```
$ man sectionnumber command
or
$ man sectionnumber filename
```

- In Oracle Solaris to look up a specific section of the man page, use the `man` command with the `-s` option, followed by the section number, and the command or file name.

```
$ man -s sectionnumber command
or
$ man -s sectionnumber filename
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The bottom portion of a man page, titled SEE ALSO, lists other commands or files related to the man page. The number in parentheses reflects the section number where the man page is located. You can use the `man` command with the `-l` option to list the man pages that relate to the same command or file name.

To view the online man page for the `passwd` file, use the following commands:

```
$ man -k passwd
...(output truncated)
passwd (1) - change user's authentication tokens
...(output truncated)
passwd (5) - password file
...(output truncated)
$ man 5 passwd
Reformatting page. Please Wait... done
File Formats passwd(5)
NAME
 passwd - password file
SYNOPSIS
 /etc/passwd
DESCRIPTION
 The file /etc/passwd is a local source of information about users'
 accounts. The password file can
 ...(output truncated)
```

## Searching the Man Pages: By Keyword

- When you are unsure of the name of a command, you can use the `man` command with the `-k` option and a keyword to search for matching man page entries.

```
$ man -k keyword
```

- The `man` command output provides a list of commands and descriptions that contain the specified keyword.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Using the `man` command, view commands containing the `calendar` keyword.

```
$ man -k calendar
cal (1) - displays a calendar
Cal (1p) - print a calendar
difftime (3p) - computes the difference between two calendar times
```

## Accessing Online Product Documentation

For additional information about Oracle products, you can access the Oracle Technical Network (OTN) Documentation website.

The screenshot shows a web browser window titled "Operating Systems Documentation" from the "docs.oracle.com" website. The page features a search bar and a sidebar with links to "Help Center Home", "Oracle Solaris", "Oracle Linux", "Oracle Solaris Cluster", "Oracle Developer Studio", and "Related Systems Software". The main content area displays two sections: "Operating Systems Documentation" with a monitor icon and "Oracle Solaris" documentation. Below these, there are sections for "Oracle Linux" and "All Oracle Linux documentation". The footer contains the Oracle logo and a copyright notice: "Copyright © 2018, Oracle and/or its affiliates. All rights reserved."

You can search the OTN website, <http://www.oracle.com/technetwork/indexes/documentation/index.html>, by subject, collection title, product category, and keyword.

Scroll down and select the Operating Systems icon.

Or for Oracle Linux documentation you can go to [https://docs.oracle.com/cd/E52668\\_01/index.html](https://docs.oracle.com/cd/E52668_01/index.html), and for Oracle Solaris documentation you can go to [http://docs.oracle.com/cd/E53394\\_01/index.html](http://docs.oracle.com/cd/E53394_01/index.html).

## Quiz



Which of the following `man` command options displays a specific section of the `man` page?

- a. -h
- b. -q
- c. -s
- d. -n



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Summary

In this lesson, you should have learned how to:

- Describe the UNIX and Linux operating systems
- Execute commands from the command line



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

2 - 54

## Practice 2: Overview

This practice covers only the highlighted topics:

- 2-1: Logging in to the system and changing your user login password
- 2-2: Displaying system information using the command line
- 2-3: Using the man pages



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

2 - 55

You will find the tasks for Practice 2 in your Activity Guide.

Unauthorized reproduction or distribution prohibited. Copyright 2017, Oracle and/or its affiliates.

Oracle University and Canadian Business School use only.

# Working with Files and Directories

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

An inode is a list of information about a particular object, such as a file, directory, or symbolic link. The metadata held by the inode includes the type of object about which the inode holds information, permissions, ownership information, and the locations in which data is stored.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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
```

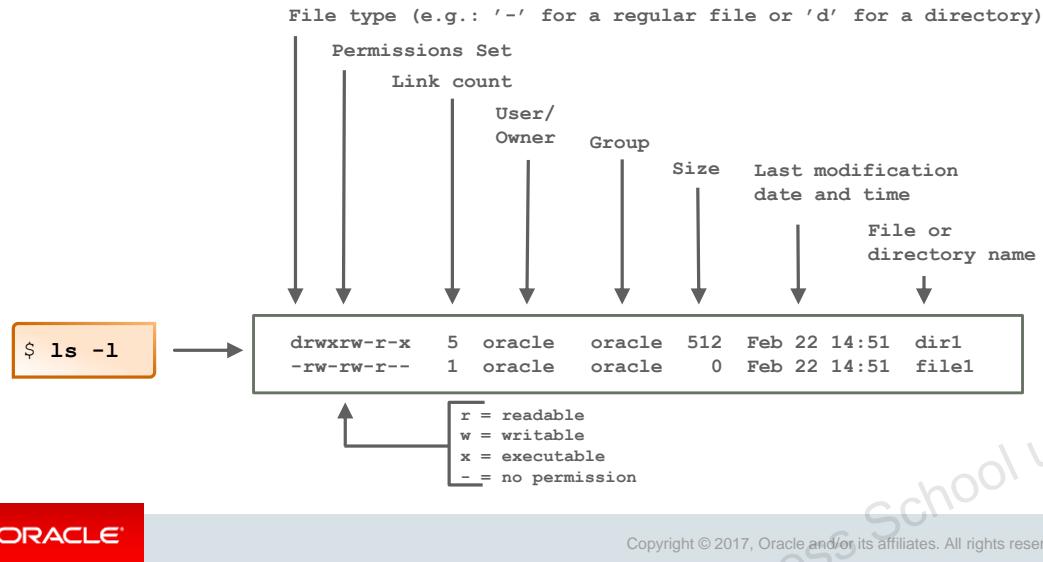


Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** For more information about `ls` command options, see the `ls` man page.

## Displaying the Directory Content with Options

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



The graphic in the slide illustrates the parts of the output of the `ls -l` command.

- The first character is the file/directory type.
- The next nine characters are the three permission sets.
- The next character represents the count of hard links to the file or directory. A hard link is a pointer, which shows the number of files or directories to which a particular file is linked within the same file system.
- The next set of characters represents the username of the user/owner (oracle).
- The next set of characters represents the group name of the group (oracle).
- The next set of characters represents the size of the file or directory in bytes.
- The next set of characters represents the time and date the file or directory was last modified.
- The last set of characters represents the name of the file or directory.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Some files are hidden from view when you use the `ls` command. Hidden files often contain information that customizes your work environment. You can use the `ls -a` command to list all files in a directory, including filenames that start with a period (.), which are hidden files.

**Note:** A single period (.) represents the current working directory. The double period (..) represents the parent directory, which contains the current working directory.

## 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`.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To obtain detailed directory information for the `dir1` directory, enter the `ls -ld` command:

```
$ ls -ld dir1
drwxr-xr-x 5 oracle oracle 512 Feb 6 09:30 dir1
```

To view a recursive list of the content of the `dir1` directory, enter the `ls -R dir1` command:

```
$ ls -R dir1
dir1:
 coffees fruit trees
 dir1/coffees:
 beans brands nuts
 dir1/coffees/beans:
 beans
 dir1/fruit:
 dir1/trees:
```

## 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



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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) |



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Knowing the file type may help you decide the command or program to use for reading the file.

**Note:** A symbolic link is a special type of file that points to another file or directory.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

The output from the `file` command is one of the following:

- Text:** Text files include American Standard Code for Information Interchange (ASCII) text, English text, command text, and executable shell scripts.
- Data:** Data files are created by programs. The `file` command indicates the type of data file, such as a FrameMaker document, if the type is known. The `file` command indicates that the file is a data file if the type is unknown.
- Executable or binary:** Executable files include 32-bit or 64-bit executable and extensible linking format (ELF) code files and other dynamically linked executable files. Executable files are commands or programs.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To change directories from the `oracle` directory to the `dir1` directory, use the `cd` command:

```
$ pwd
/home/oracle
$ cd lab/dir1
$ pwd
/home/oracle/lab/dir1
```

The `cd` command is a bash built-in command.

## 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 |



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To change directories using a relative pathname, enter the `cd` command with the pathname that starts from the current working directory, `oracle`.

```
$ cd ~/lab
$ cd dir1
$ pwd
/home/oracle/lab/dir1
$ cd ../dir2
$ pwd
/home/oracle/lab/dir2
$ cd ..
$ cd dir1/coffees
$ pwd
/home/oracle/lab/dir1/coffees
```

To change directories using an absolute pathname, enter the `cd` command with the complete pathname from the root (/) directory.

```
$ cd
$ cd /home/oracle/lab/dir1/coffees
$ pwd
/home/oracle/lab/dir1/coffees
```

## 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`.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** You can configure systems to use the `/home` directory, instead of the `/export/home` directory, as the default home directory.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To navigate to a user’s home directory, enter the `cd` command with a tilde (~) symbol in front of the username. The tilde (~) metacharacter equates to the absolute pathname of the user’s home directory.

**Note:** The tilde (~) shell metacharacter is not available in all shells.

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

You can use the tilde (~) symbol to represent your home directory in a relative path. The tilde (~) in the following example represents the `oracle` home directory:

```
$ cd ~/lab/dir1/fruit
$ pwd
/home/oracle/lab/dir1/fruit
```

You can also use the tilde (~) symbol to navigate to another user’s home directory.

```
$ cd ~user2
$ pwd
/home/user2
$ cd
$ pwd
/home/oracle
```

## Quiz



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

- a. True
- b. False



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To use the `cat` command to concatenate several text files and redirect them to another text file.

```
$ cat first_file second_file > results_file
```

**Caution:** Do not use the `cat` command to read binary files. Using the `cat` command to read binary files can cause a terminal window to freeze. If your terminal window freezes, close the terminal window, and open a new terminal window.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

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



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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 [-[+]aABcCdeEfFgGiIJKLMNOPQRSTUVWXYZ]
 [-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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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`.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To display the last four lines of the `/usr/dict/words` file, enter the `tail` command with the `-n` option set to 4.

```
[oracle@s11-server1:~]$ tail -4 /usr/dict/words
zounds
z's
zucchini
Zurich
```

To display line 25136 through the end of the `/usr/dict/words` file, enter the `tail` command with the `+n` option set to 25136.

```
[oracle@s11-server1:~]$ tail +25136 /usr/dict/words
Zorn
Zoroaster
Zoroastrian
zounds
z's
zucchini
Zurich
```

**Note:** For Oracle Linux, the file is `/usr/share/dict/words`.

## 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     |
|-----------------|-----------------|
| <code>-l</code> | Line count      |
| <code>-w</code> | Word count      |
| <code>-c</code> | Byte count      |
| <code>-m</code> | Character count |



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** When you use the `wc` command without options, the output displays the number of lines, words, and characters contained in the file.

To display the number of lines, words, and characters in the `dante` file, use the `wc` command.

```
$ wc dante
32 223 1319 dante
```

To display the number of lines in the `dante` file, enter the `wc` command with the `-l` option.

```
$ wc -l dante
32 dante
```

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

For illustration purposes, let's make a copy of the `dante` file as `copy_dante`. Then change the mixed case "Dante" on line 1 of the `copy_dante` file to all caps "DANTE". An `ls -l` shows that the files are the exact same size. The question is: Is there any difference between the two files?

```
$ cp dante copy_dante
$ vi copy_dante
$ ls -l dante copy_dante
-rw-rw-r--. 1 oracle oracle 1319 ... copy_dante
-rw-rw-r--. 1 oracle oracle 1319 ... dante
$ diff dante copy_dante
< The Life and times of Dante

> The Life and times of DANTE
```

## Quiz



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

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



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



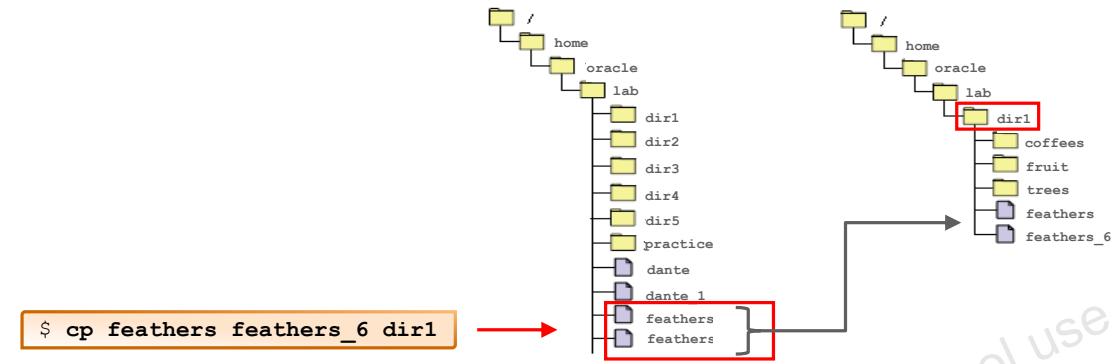
Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

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

```
$ cd ~/lab
$ pwd
/home/oracle/lab
$ ls
dante dir3 file.2 file3 greetings
dante_1 dir4 file.3 file4 myvars
dir1 dir5 file1 fruit practice
dir2 file.1 file2 fruit2 tutor.vi
$ cp file3 feathers
$ ls
dante dir3 file.1 file2 fruit2 tutor.vi
dante_1 dir4 file.2 file3 greetings
dir1 dir5 file.3 file4 myvars
dir2 feathers file1 fruit practice
```

## 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.



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To copy the `feathers` and `feathers_6` files from the student directory into the `dir1` subdirectory, enter the following commands:

```
$ pwd
/home/oracle/lab
$ ls dir1
coffees fruit trees
$ cp feathers feathers_6 dir1
$ ls dir1
coffees feathers feathers_6 fruit trees
```

## Copying Files: `cp` Command Options

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

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



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To copy the contents of the `dir3` directory to a new directory named `dir10`, use the `cp -r` command. Both directories are in the `oracle/lab` directory.

```
$ cd
$ pwd
/home/oracle/lab
$ ls dir3
planets
$ cp dir3 dir10
cp: dir3: is a directory
$ cp -r dir3 dir10
$ ls dir10
planets
$ ls dir3
planets
```

## 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 `yes/no` 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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/rename 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** Unlike the `cp` command, the `mv` command moves the respective file or directory, and the original no longer exists.

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

The `-i` option prompts you for confirmation to prevent you from automatically overwriting existing files with the `mv` command.

`$ mv -i source target`

- A `yes` response permits the `mv` command to overwrite the existing files.
- A `no` response prevents the `mv` command from overwriting the existing files.

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** When you move a single directory to a target directory that does not exist, you actually rename the current directory and its path. When you move multiple directories to a target directory that does not exist, the following error message appears:

`mv: target_directory not found.`

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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`



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To create an empty file named `space` in the `dir3` directory, enter the following commands:

```
$ pwd
/home/oracle/lab
$ cd dir3
$ ls
planets
$ touch space
$ ls
planets space
```

Use the `touch` command to create three empty files named `moon`, `sun`, and `cosmos` in the `dir3` directory.

```
$ touch moon sun cosmos
$ ls
cosmos moon planets space sun
```

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** The command used with the `-p` option creates all of the nonexisting parent directories that do not yet exist in the path to the new directory.

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

To create a new directory named empty\_directory located inside a directory named newdir, use the `mkdir` command with the `-p` option. The `newdir` directory does not yet exist.

```
$ cd ~/lab
$ pwd
/home/oracle/lab
$ mkdir -p newdir/empty_directory
$ ls newdir
empty_directory
```

To create the `dir1`, `dir2`, and `dir3` directories in the `Weekly` directory, enter the `mkdir` command:

```
$ cd Reports/Weekly
$ mkdir dir1 dir2 dir3
$ ls -F
dir1/ dir2/ dir3/
```

## 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                                                                                                        |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| <code>-r</code> or <code>-R</code> | Recursive, includes the contents of a directory and the contents of all subdirectories when you remove a directory |
| <code>-i</code>                    | Prevents the accidental removal of existing files or directories by prompting for a 'yes' or a 'no'                |
| <code>-f, --force</code>           | Forces the removal of existing files and directories (the opposite of <code>-i</code> ) without prompting          |

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



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

- The `-r` or `-R` option allows you to remove directories that contain files and subdirectories.
- The `-i` option prompts you for confirmation before removing any file.
  - A yes response completes the removal of the file.
  - A no response aborts the removal of the file.

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** The asterisk (\*) symbol used in the second example in the slide is a wildcard character, sometimes called a “glob.”

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** If you do not use the `-r` or `-R` option with the `rm` command while removing directories, the following error message appears: `rm: directoryname: is a directory`.

To interactively remove a directory and its contents, use the `-i` option along with the `rm -r` command. Create a new directory called `rmtest` in your `/home/oracle/lab` directory by using the `rm -ir` command.

```
$ rm -ir rmtest
rm: examine files in directory rmtest (yes/no)? y
rm: remove rmtest/testfile (yes/no)? y
rm: remove rmtest: (yes/no)? y
$ ls
Reports dir10 feathers file1 fruit2
brands dir2 feathers_6 file2 greetings
dante dir3 file.1 file3 myvars
dante_1 dir4 file.2 file4 newdir
dir1 dir5 file.3 fruit tutor.vi
```

## 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 ..`).



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** You use the `rmdir` command to remove empty directories. You use the `rm` command with the `-r` option to remove directories that contain files and subdirectories.

Remove `empty_directory` by using the `rmdir` command:

```
$ cd ~/lab
$ pwd
/home/oracle/lab
$ cd newdir
$ pwd
/home/oracle/lab/newdir
$ ls -F
empty_directory/
$ rmdir empty_directory
$ ls
```

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

In the syntax displayed in the slide, the `source` variable refers to the file to which you create the link. The `target` variable refers to the name of the symbolic link. When creating a symbolic link, if the `source` does not exist, a symbolic link, which points to a nonexistent file, is created.

## 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/ dir1/ feathers file1* fruit2
brands dir2/ feathers_6 file2* greetings
dante_1 dir3/ file.1* file3* myvars
dante_link@ dir4/ file.2* file4*
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** The @ symbol that follows the filename indicates that the file is a symbolic link. The output of the `ls -F` command in the slide lists the `dante_link` file as a symbolic link.

To further view the content of the `dante_link` file, use the `cat` command.

```
$ cat dante_link
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. For Dante
Alighieri, like Raphael, Michelangelo, Galileo, etc. is usually
referred to by his first name
... (output truncated)
```

To see the pathname to which a symbolic link is pointing to, enter the `ls -l` command with the symbolic link file name.

```
$ ls -l dante_link
lrwxrwxrwx 1 oracle oracle ... 14:17 dante_link -> /var/tmp/dante
```

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

In the syntax displayed in the slide, the `source` variable refers to the file to which you create the link. The `target` variable refers to the name of the hard link. When creating a hard link, if the `source` does not exist, the create fails and in Oracle Solaris displays “`ln: cannot access <source>`,” and in Oracle Linux displays “`ln: failed to access '<source>': no such file or directory.`”

Hard links can be used to make a backup of the original file. Thus if someone tries to move or remove the original file, the backup would remain.

**Note:** This will not protect the original file from saved editing changes.

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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`



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Quiz

The `rm -r` command removes nonempty directories.

- a. True
- b. False



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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*.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

For more information about regular expressions, in Oracle Linux, use `man 7 regex` to view the man pages, and in Oracle Solaris, use `man -s 5 regex` to view the man pages.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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*
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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 {} \;
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Note:** The `locate` command is dependent on the system administrator scheduling an `updatedb` command, which updates the location database. Usually, the scheduling of the `updatedb` command is only once daily. Therefore, the files you just created might not be found by `locate` until the following day.

## 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.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

In addition, originally there were two variant programs, `egrep` and `fgrep`. Currently, `egrep` has been replaced by `grep -E` and `fgrep` has been replaced by `grep -F`.

- `grep -E`: Interprets *pattern* as an **extended-regexp** regular expression
- `grep -F`: Interprets *pattern* as a list of **fixed-strings**, separated by newlines, any of which is to be matched

**Note:** Both `egrep` and `fgrep` may still be available.

## 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
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## 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



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 3 in your Activity Guide.

Unauthorized reproduction or distribution prohibited. Copyright 2017, Oracle and/or its affiliates.

Oracle University and Canadian Business School use only.

# Using the vim Editor

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Objectives

After completing this lesson, you should be able to:

- Access the `vim` editor
- Modify files with the `vim` editor



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Agenda

- Accessing the vim Editor
- Modifying Files with the vim Editor



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## vim Editor: Introduction

- The `vim` editor is an interactive command-line editor that you can use to create and modify text files.
  - The `vim` editor is also the only text editor that you can use to edit certain system files without changing the permissions associated with the files.
- In Oracle Solaris and Oracle Linux, `vim` (`vi` improved) is the default editor.
  - The `vim` editor is an enhanced version of the `vi` editor and is accessed via an alias “`vi`” in Oracle Linux.
  - In Oracle Solaris, `vi` is a symbolic link that links to `vim`.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** This lesson focuses mainly on the `vim` editor. For additional information about the `vim` editor, use the `vimtutor` command on the command prompt to go through the built-in tutorial for beginners. You can also access the *Vim Users' Manual* that details the features of Vim. This too is available from within `vim`, or can be found online.

## Accessing the `vim` Editor

- To create, edit, and view files in the `vim` editor, use the `vi` command.
- The `vi` command includes the following three syntaxes:

```
$ vi [options] filename
$ vi
$ vi filename
```

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

If the system crashes while you are editing a file, you can use the `-r` option to recover the file.

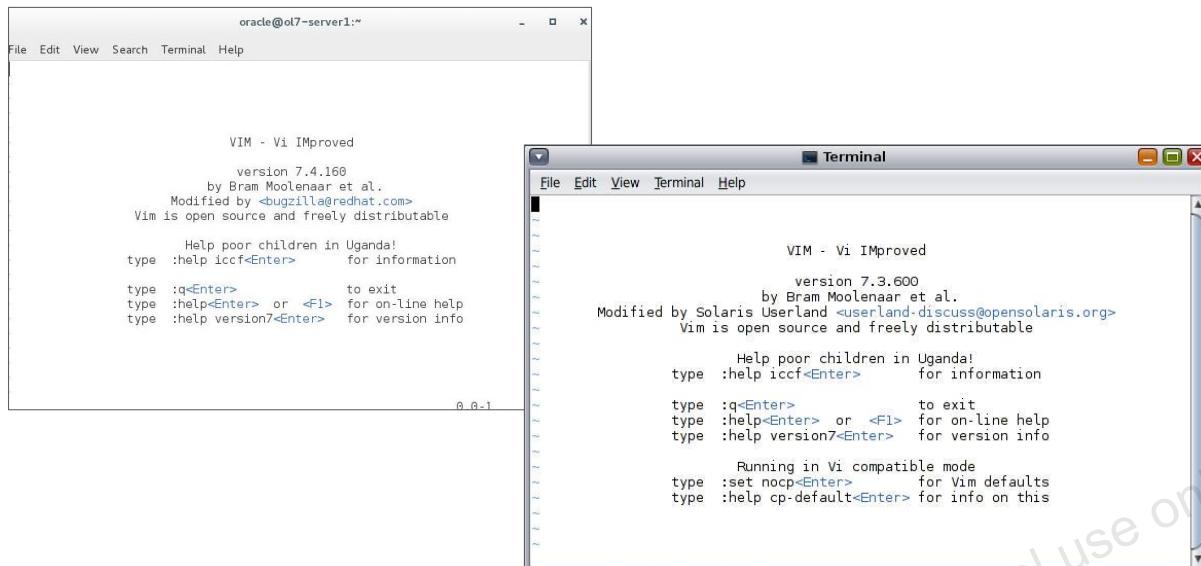
```
$ vi -r filename
```

The file opens so that you can edit it. You can then save the file and exit the `vi` editor, by using the following command inside `vim`, press Esc followed by `:wq`.

```
$ vi -R filename
```

Opens the file in read-only mode to prevent accidental overwriting of the contents of the file.

## vim Editor: Overview



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The first screen of the `vim` editor in a terminal window is a blank window filled with tildes (~) and a blinking cursor in the top-left corner. The screenshots displayed in the slide are those of the `vim` editor in Oracle Linux (upper-left corner) and Oracle Solaris (lower-right corner).

## vim Editor Modes

- The vim editor is a modal editor and provides six basic modes of operation:
  1. Command mode (normal): Typically the default mode when vim starts
  2. Insert (and replace) mode: Used when adding new text
  3. Visual mode: Changes how text is highlighted
  4. Select mode: Is the default mode for MS-Windows installations
  5. Command-line mode: Allows you enter ex commands
  6. Ex mode: Is optimized for batch processing
- The command and insert modes are the two most common modes and will be the ones used in this course.

**Note:** Originally the vi editor was the **visual interface** (vi) to the ex editor, which in turn is an **extended** version of the ed editor.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The vim editor provides two common modes of operation:

**Command mode:** Command mode is the default mode for the vim editor. In this mode, you can run commands to delete, change, copy, and move text. You can also position the cursor, search for text strings, and exit the vim editor.

**Insert mode:** You can insert text into a file in insert mode. The vim editor interprets everything you type in insert mode as text. To invoke insert mode, press one of the following keys:

- i – Inserts text before the cursor
- o – Opens a new blank line below the cursor
- a – Appends text after the cursor
- R – Replaces text after the cursor

You can also invoke insert mode to insert text into a file by pressing one of the following uppercase keys:

- I – Inserts text at the beginning of the line
- O – Opens a new blank line above the cursor
- A – Appends text at the end of the line

## Switching Between the Two Most Common Modes

- The default mode for the `vim` editor is **command mode**.
- To switch to **insert mode**, press `i`, `o`, `a` or `R`.
- To return to **command mode**, press the **Esc** key.
- In command mode, enter the `:wq` command that writes (saves changes to the file), quits the `vim` editor, and returns to the shell prompt.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Agenda

- Accessing the vim Editor
- Modifying Files with the vim Editor



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Viewing Files in Read-Only Mode

- The `view` command enables you to view files in **read-only** mode.

```
$ view filename
or
$ vim -R filename
```

- The `view` command invokes the `vim` editor as read-only, which means you cannot save changes to the file.
- To view the `dante` file in read-only mode, enter the following command:

```
$ view dante
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Moving the Cursor Within the `vim` Editor

| Key Sequence                        | Cursor Movement                                    |
|-------------------------------------|----------------------------------------------------|
| <b>H, left arrow, or Backspace</b>  | Left one character                                 |
| <b>J or down arrow</b>              | Down one line                                      |
| <b>K or up arrow</b>                | Up one line                                        |
| <b>L, right arrow, or space bar</b> | Right (forward) one character                      |
| <b>W</b>                            | Forward one word                                   |
| <b>B</b>                            | Back one word                                      |
| <b>E</b>                            | To the end of the current word                     |
| <b>\$</b>                           | To the end of the line                             |
| <b>0 (zero)</b>                     | To the beginning of the line                       |
| <b>^</b>                            | To the first non-white space character on the line |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide shows the key sequences that move the cursor in the `vim` editor.

## Moving the Cursor Within the `vim` Editor

| Key Sequence          | Cursor Movement                        |
|-----------------------|----------------------------------------|
| <b>Return / Enter</b> | Down to the beginning of the next line |
| <b>G</b>              | To the last line of the file           |
| <b>1G</b>             | To the first line of the file          |
| <b>:n</b>             | To Line n                              |
| <b>nG</b>             | To Line n                              |
| <b>Control + F</b>    | Pages forward one screen               |
| <b>Control + D</b>    | Scrolls down one half screen           |
| <b>Control + B</b>    | Pages back one screen                  |
| <b>Control + U</b>    | Scrolls up one half screen             |
| <b>Control + L</b>    | Refreshes the screen                   |
| <b>Control + G</b>    | Displays current buffer information    |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide shows the key sequences that move the cursor in the `vim` editor.

## Inserting and Appending Text

The table describes the commands to insert and append text to a new or existing file by using the `vim` editor.

| Command                  | Function                                                                                                  |
|--------------------------|-----------------------------------------------------------------------------------------------------------|
| <code>a</code>           | Appends text after the cursor                                                                             |
| <code>A</code>           | Appends text at the end of the line                                                                       |
| <code>i</code>           | Inserts text before the cursor                                                                            |
| <code>I</code>           | Inserts text at the beginning of the line                                                                 |
| <code>o</code>           | Opens a new line below the cursor                                                                         |
| <code>O</code>           | Opens a new line above the cursor                                                                         |
| <code>:r filename</code> | Reads and inserts the contents of another file into the current file below the line containing the cursor |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The `vim` editor is case-sensitive. Use the appropriate case for the insert commands. Also, most of the insert commands and cursor movements can be preceded by a number to repeat the command that many times.

## Text-Deletion Commands

The table shows commands that delete text in the `vim` editor.

| Command            | Function                                                                                                                    |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <code>R</code>     | Overwrites or replaces characters on the line at and to the right of the cursor. To terminate this operation, press Escape. |
| <code>C</code>     | Changes or overwrites characters from the cursor to the end of the line                                                     |
| <code>s</code>     | Substitutes a string for a character at the cursor                                                                          |
| <code>x</code>     | Deletes a character at the cursor                                                                                           |
| <code>nx</code>    | Deletes n characters beginning at the cursor                                                                                |
| <code>dw</code>    | Deletes a word or part of the word to the right of the cursor                                                               |
| <code>dd</code>    | Deletes the line containing the cursor                                                                                      |
| <code>ndd</code>   | Deletes n lines beginning with the line containing the cursor                                                               |
| <code>D</code>     | Deletes the line from the cursor to the right end of the line                                                               |
| <code>:n,nd</code> | Deletes lines n–n. For example, <code>:5,10d</code> deletes lines 5–10.                                                     |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You can use several commands to edit files by using the `vim` editor. The following sections describe basic operations for deleting, changing, replacing, copying, and pasting. Remember that the `vim` editor is case-sensitive.

**Note:** Output from the delete command writes to a buffer from which text can be retrieved.

## Edit Commands

The table describes the commands to change text, undo a change, and repeat an edit function in the `vim` editor.

| Command         | Function                                                                            |
|-----------------|-------------------------------------------------------------------------------------|
| <code>cw</code> | Changes or overwrites characters at the cursor location to the end of that word     |
| <code>r</code>  | Replaces the character at the cursor with one other character                       |
| <code>J</code>  | Joins the current line and the line below                                           |
| <code>xp</code> | Transposes the character at the cursor and the character to the right of the cursor |
| <code>~</code>  | Changes letter casing to uppercase or lowercase, at the cursor                      |
| <code>u</code>  | Undoes the previous command                                                         |
| <code>U</code>  | Undoes all changes to the current line                                              |
| <code>.</code>  | Repeats the previous command                                                        |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** Many of these commands change the `vim` editor into the insert mode. To return to the command mode, press the Esc key.

## Quiz



In which vim mode are commands normally initiated?

- a. ed mode
- b. ex mode
- c. Command mode
- d. Input mode



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Searching for and Substituting (Replacing) Text Within a File

The table shows the commands that search for and can substitute (replace) text using the vim editor.

| Command                           | Function                                                                                                                                                                                                   |
|-----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| /pattern                          | Searches forward for the pattern/string ( <i>regex_pattern</i> ).                                                                                                                                          |
| ?pattern                          | Searches backward for the pattern/string ( <i>regex_pattern</i> ).                                                                                                                                         |
| n                                 | Searches for the next occurrence of the pattern. Use this command after searching for a pattern.                                                                                                           |
| N                                 | Searches for the previous occurrence of the pattern. Use this command after searching for a pattern.                                                                                                       |
| :[%]s/oldstring/newstring<br>/[g] | Searches for the old string and substitutes (replaces) it with the new string globally. The “%” symbol searches the whole file and the “g” replaces every occurrence of oldstring with newstring globally. |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Hypnotically, what if you wanted to substitute (*s*) the occurrences of the search string “Dante” in the dante text file and replace it with the string “Homer”. For illustration purposes, I have also added a second “Dante” to the end of line 7, that makes for 6 occurrences of the string “Dante” in this abbreviated file.

```
$ cat -n first10_dante
 1 The life and Times of Dante
 2
 3 by Dante Pacai
 4
 5
 6 Mention “Alighieri” and few may know about whom you are talking. Say
 7 “Dante,” instead, and the whole world knows whom you mean. For Dante
 8 Dante Alighieri, like Raphael, Michelangelo, Galileo, etc. is usually
 9 referred to by his first name. There is only one Dante, as we
 recognize
10 One Raphael, one Michelangelo, and one Galileo.
```

From within vim you can do a search and replace of a *regex\_pattern*. Here are four examples:

- :%s/Dante/Homer/ conformation 5 substitutions on 5 lines.  
**Note:** The second occurrence of “Dante” on line 7 did not get substituted.
- However, adding a *g* to the end of the substitution command :%s/Dante/Homer/g conformation 6 substitutions on 5 lines, changes all 6 occurrences of “Dante”.
- To change only the 2 occurrences of “Dante” at the end of the line, add a \$ to the search string. :%s/Dante\$/Homer/ changes the occurrences at the end of line 1 and line 7.
- To change only the occurrence “Dante” at the beginning of the line, add a ^ to the search string. :%s/^Dante/Homer/ changes the only occurrence at the beginning of line 8.

## Copy-and-Paste Commands

The table shows the commands that copy (`yank`) and paste (`put`) text in the `vim` editor.

| Command                     | Function                                                                                                                      |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <code>yy</code>             | Yanks a copy of a line containing the cursor                                                                                  |
| <code>n<sup>y</sup>y</code> | Yanks a copy of n lines, beginning with the line containing the cursor                                                        |
| <code>p</code>              | Puts yanked (yy) or deleted (dd) text after the line containing the cursor                                                    |
| <code>P</code>              | Puts yanked (yy) or deleted (dd) text before the line containing the cursor                                                   |
| <code>:n,n co n</code>      | Copies lines n–n and puts them after line n. For example, <code>:1,3 co 5</code> copies lines 1–3 and puts them after line 5. |
| <code>:n,n m n</code>       | Moves lines n–n to line n. For example, <code>:4,6 m 8</code> moves lines 4–6 to after line 8.                                |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Save and Quit Commands

The table describes the commands that save (write) the text file, quit the `vim` editor, and return to the shell prompt.

| Command                          | Function                                            |
|----------------------------------|-----------------------------------------------------|
| <code>:w</code>                  | Saves the file with changes by writing to the disk  |
| <code>:w<br/>new_filename</code> | Writes the contents of the buffer to new_filename   |
| <code>:wq</code>                 | Saves the file with changes and quits the vi editor |
| <code>:x</code>                  | Saves the file with changes and quits the vi editor |
| <code>zz</code>                  | Saves the file with changes and quits the vi editor |
| <code>:q!</code>                 | Quits without saving changes                        |
| <code>zQ</code>                  | Quits without saving changes                        |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Session Customization

- You can customize a `vim` session by setting options for the session.
- In VIM 7.n, there are more than 295 options.
- When you set an option, you enable a feature that is not activated by default.
- You can use the `:set` command to enable and disable options within `vim`.
- Two of the many `set` command options include displaying line numbers and invisible characters, such as the Tab and the end-of-line (^M) characters.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

There are two files that the `vim` editor reads to customize your editing session. For legacy systems that had used the `vi` editor, the `.exrc` file is used. For systems using the `vim` editor, the file is `.vimrc`. However, the `vim` editor reads whichever file is there.

To create an automatic customization for all your vim sessions, perform the following steps:

1. Create either a file named `.exrc` or `.vimrc` in your home directory.
2. Enter any of the `set` variables into either file.
3. Enter each `set` variable without the preceding colon.
4. Enter each command on one line.

The `vim` editor reads whichever of the two files is located in your home directory each time you open a `vim` session, regardless of your current working directory. You can provide overrides by having a second `.vimrc` in the current working directory.

In the next two slides there is an example `.vimrc` (`vimrc-example.vim`) text file delivered with both Oracle Linux and Oracle Solaris. The full directory path name for Oracle Linux is `/usr/share/vim/vim74` and the full directory path name for Oracle Solaris is `/usr/share/vim/vim73`. The file contains: How to use, descriptions of many options, and overall documentation.

Also, you can customize `vim` for editing different programming languages. See [www.vim.org](http://www.vim.org).

```
$ cat /usr/share/vim/vim7[34]/vimrc_example.vim
" An example for a vimrc file.
"
" Maintainer: Bram Moolenaar <Bram@vim.org>
" Last change: 2011 Apr 15
"
" To use it, copy it to
" for Unix and OS/2: ~/.vimrc
" for Amiga: s:.vimrc
" for MS-DOS and Win32: $VIM_vimrc
" for OpenVMS: sys$login:.vimrc
" When started as "evim", evim.vim will already have done these settings.
if v:progname =~? "evim"
 finish
endif
" Use Vim settings, rather than Vi settings (much better!).
" This must be first, because it changes other options as a side effect.
set nocompatible
" allow backspacing over everything in insert mode
set backspace=indent,eol,start
if has("vms")
 set nobackup " do not keep a backup file, use versions instead
else
 set backup " keep a backup file
endif
set history=50 " keep 50 lines of command line history
set ruler " show the cursor position all the time
set showcmd " display incomplete commands
set incsearch "do incremental searching
" For Win32 GUI: remove 't' flag from 'guioptions': no tearoff menu entries
" let &guioptions = substitute(&guioptions, "t", "", "g")
" Don't use Ex mode, use Q for formatting
map Q qq
" CTRL-U in insert mode deletes a lot. Use CTRL-G u to first break undo,
" so that you can undo CTRL-U after inserting a line break.
inoremap <C-U> <C-G>u<C-U>
" In many terminal emulators the mouse works just fine, thus enable it.
if has('mouse')
 set mouse=a
endif
```

```
" Switch syntax highlighting on, when the terminal has colors
" Also switch on highlighting the last used search pattern.
if &t_Co > 2 || has("gui_running")
 syntax on
 set hlsearch
endif
" Only do this part when compiled with support for autocmds.
if has("autocmd")
 " Enable file type detection.
 " Use the default filetype settings, so that mail gets 'tw' set to 72,
 " 'cindent' is on in C files, etc.
 " Also load indent files, to automatically do language-dependent
indenting.
 filetype plugin indent on
 " Put these in an autocmd group, so that we can delete them easily.
augroup vimrcEx
au!
 " For all text files set 'textwidth' to 78 characters.
autocmd FileType text setlocal textwidth=78
 " When editing a file, always jump to the last known cursor position.
 " Don't do it when the position is invalid or when inside an event
handler
 " (happens when dropping a file on gvim).
 " Also don't do it when the mark is in the first line, that is the
default
 " position when opening a file.
autocmd BufReadPost *
 \ if line("'"") > 1 && line("'"") <= line("$") |
 \ exe "normal! g`'""
 \ endif
augroup END
else
 set autoindent " always set autoindenting on
endif " has("autocmd")
" Convenient command to see the difference between the current buffer and
the
" file it was loaded from, thus the changes you made.
" Only define it when not defined already.
if !exists(":DiffOrig")
 command DiffOrig vert new | set bt=nofile | r ++edit # | 0d_ | diffthis
 \ | wincmd p | diffthis
endif
```

## Session Customization Commands

| Command                      | Function                                                                                                        |
|------------------------------|-----------------------------------------------------------------------------------------------------------------|
| <code>:set nu</code>         | Shows line numbers                                                                                              |
| <code>:set nonu</code>       | Hides line numbers                                                                                              |
| <code>:set ic</code>         | Instructs searches to ignore case                                                                               |
| <code>:set noic</code>       | Instructs searches to be case-sensitive                                                                         |
| <code>:set list</code>       | Displays invisible characters, such as <code>^I</code> for a Tab and <code>\$</code> for end-of-line characters |
| <code>:set nolist</code>     | Turns off the display of invisible characters                                                                   |
| <code>:set showmode</code>   | Displays the current mode of operation                                                                          |
| <code>:set noshowmode</code> | Turns off the mode of operation display                                                                         |
| <code>:set</code>            | Displays all the <code>vim</code> variables that are set                                                        |
| <code>:set all</code>        | Displays all <code>vim</code> variables and their current values                                                |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide describes some of 295 options of the `set` command.

## Quiz



Which three commands help save changes in your file and quit the `vim` editor?

- a. `:wq`
- b. `:wq!`
- c. `ZZ`
- d. `:q!`
- e. `:w`



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Quiz



Which of the following commands searches backwards for the pattern?

- a. ?pattern
- b. /pattern
- c. !pattern
- d. ~pattern



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Summary

In this lesson, you should have learned how to:

- Access the `vim` editor
- Modify files with the `vim` editor



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Practice 4: Overview

- 4-1: Using the `vim` Editor



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 4 in your Activity Guide.

Unauthorized reproduction or distribution prohibited. Copyright 2017, Oracle and/or its affiliates.

Oracle University and Canadian Business School use only.

# Using Features Within the Bash Shell

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Objectives

After completing this lesson, you should be able to:

- Use shell expansion for generating shell tokens
- Use shell metacharacters for command redirection
- Use variables in the `bash` shell to store values
- Display the command history
- Customize the user's work environment



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Lesson Agenda

- Using Shell Expansion for Generating Shell Tokens
- Using Shell Metacharacters for Command Redirection
- Using Variables in the bash Shell to Store Values
- Displaying the Command History
- Customizing the User's Work Environment



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Shell Expansions

- While working in a shell, sets or ranges of information are often repeated.
- Shell expansions help generate a large number of shell tokens by using compact syntaxes.
- Expansion is performed on the command line after the command is split into tokens.
- Some of the more common types of shell expansions are:
  - Brace expansion
  - Tilde expansion
  - Parameter expansion
  - Command substitution
  - Path name expansion/file name generation



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Brace Expansion

- The brace ({} ) expansion is a mechanism by which arbitrary strings may be generated.
- Patterns to be brace-expanded take the form of an optional preamble, followed by either a series of comma-separated strings or a sequence expression between a pair of curly braces, followed by an optional postscript.

```
Brace expansion syntax: optional preamble{string1[,string2][,stringn]}optional postscript
```

- In this syntax, the **preamble** “a” is prefixed to each string contained within the braces, and the **postscript** “e” is then appended to each resulting string, expanding left to right.

```
$ echo a{d,c,b}e
ade ace abe
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Brace expansion can also be used to create a list or range of values using the syntax “beginning .. ending”.

```
$ echo {3..7}
3 4 5 6 7
```

## Tilde Expansion

The tilde expansion includes:

- The tilde (~) symbol, which represents the home directory of the current user
- The tilde (~) symbol with a username, which represents the home directory of the specified user



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The tilde (~) symbol is a metacharacter that equates to the absolute path name of the user's home directory.

To change directories to `lab/dir1` in the user's home directory by using the tilde (~) symbol:

```
$ cd ~/lab/dir1
$ pwd
/home/oracle/lab/dir1/
```

Change directories to the `user2` home directory using the tilde (~) symbol followed by a username:

```
$ cd ~user2
$ pwd
/home/user2
```

## Parameter Expansion

- In UNIX and Linux, there can be hundreds of parameters/variables.
- The parameter expansion includes:
  - The dollar sign (\$) symbol
- The following example shows just two variables, `USER` and `HOME`:

```
$ echo $USER
oracle
$ echo $HOME
/home/oracle
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Command Substitution

- Command substitution allows you to use the output of a command as an expression to another command (much like running a command within a command).
- The command substitution includes:
  - Dollar sign and a pair of open/close round bracket (\$( )) symbols (\$ (command))
  - A pair of backquotes (backticks) (` command `)
- Use the ls -l to list an executable file, when you do not know which directory it is in. Start with the which command to locate the file and then use command substitution to complete the process.

```
$ which passwd
/usr/bin/passwd
$ ls -l $(which passwd)
-rwsr-xr-x 1 oracle oracle ... passwd
or
$ ls -l `which passwd`
-rwsr-xr-x 1 oracle oracle ... passwd
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Path Name Expansion and File Name Generation

- The path name expansion simplifies location changes within the directory hierarchy.
- The path name expansion or file name generation includes:
  - The asterisk (\*) symbol, which matches zero or more characters (sometimes called “globbing”)
  - The question mark (?) symbol, which matches zero or a single character
  - A pair of square brackets ([ ]), which matches a single character
  - The dash (-) symbol, which represents the previous working directory

**Note:** The asterisk, question mark, and square brackets are metacharacters also used by regular expressions.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Switch between the `user1` and `tmp` directories using the dash (-) expansion symbol.

```
$ cd
$ pwd
/home/user1
$ cd /tmp
$ pwd
/tmp
$ cd -
/home/user1
$ cd -
/tmp
```

In the user's home directory, use the asterisk (\*) expansion symbol.

```
$ echo D*
Documents Desktop
```

## Asterisk (\*) Expansion Symbol

- The asterisk (\*) expansion symbol is also a wildcard character or glob, and matches zero or more characters, except the leading period (.) of a hidden file.
- List all files and directories that start with the letter `f` followed by zero or more other characters.

```
$ cd
$ ls f*
feathers file.1 file.2 file.3 file4 fruit2
feathers_6 file1 file2 file3 fruit
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Using the asterisk expansion symbol, list all files and directories that end with the number 3, preceded by zero or more characters.

```
$ ls *3
file.3 file3
dir3:
cosmos moon planets space sun vegetables
```

## Question Mark (?) Expansion Symbol

- The question mark (?) expansion symbol is also a wildcard character and matches any single character, except the leading period (.) of a hidden file.
- List all files and directories that start with the string `dir` and followed by one other character.

```
$ ls dir?
dir1:
 coffees fruit trees

dir2:
 beans notes recipes

dir3:
 cosmos moon planets space sun vegetables
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** If no files match an entry using the question mark (?) expansion symbol, an error message appears.

```
$ ls z?
z?: No such file or directory
```

## Square Bracket ( [ ] ) Expansion Symbols

- The square bracket ( [ ] ) expansion symbols are used to create a *character class*, which represents a set or range of characters for a *single* character position.
  - A set of characters is any number of specific characters, for example, [acb].
    - The characters in a set do not necessarily have to be in any order, for example, [abc] is the same as [cab].
  - A range of characters is a series of ordered characters.
    - A range lists the first character followed by a hyphen (-) and then the last character, for example, [a-z] or [0-9].
    - When specifying a range, arrange the characters in the order that you want them to appear in the output, for example, use [A-Z] or [a-z] to search for any uppercase or lowercase alphabetical character, respectively.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

List all files and directories that start with the range letters a through f.

```
$ ls [a-f]*
brands dante_1 file.1 file2 file4
celery feathers file1 file.3 fruit
dante feathers_6 file.2 file3 fruit2
dir1:
coffees fruit trees
```

List all files and directories that start with only the letters f or p.

```
$ ls [fp]*
feathers file.1 file.2 file.3 file4
fruit2
feathers_6 file1 file2 file3 fruit
perm:
group motd skel vfstab
practice1:
appointments file.1 file.2 play
```

## Quiz



Which of the following expansion symbols equates to the absolute path name of the user's home directory?

- a. #
- b. [ ]
- c. \*
- d. ~



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Lesson Agenda

- Using Shell Expansion for Generating Shell Tokens
- Using Shell Metacharacters for Command Redirection
- Using Variables in the bash Shell to Store Values
- Displaying the Command History
- Customizing the User's Work Environment



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Shell Metacharacters

- Shell metacharacters are specific characters, generally symbols, that have special meaning within the shell.
- bash metacharacters:
  - | pipe, sends the output of the command on the left as input to the command on the right of the symbol.
  - & ampersand, background execution
  - ; semicolon, command separator
  - \ backslash, escapes the next metacharacter to remove its meaning.
  - ( ) round brackets (parentheses), command grouping
  - < > >> & angle brackets (less-than and greater-than), redirection symbols
  - ` ` \$ ( ) backquote (backtick), command substitution
  - space tab newline “whitespace” Internal Field Separator (IFS)

**Note:** The subsequent slides on this topic cover only the redirection symbols.

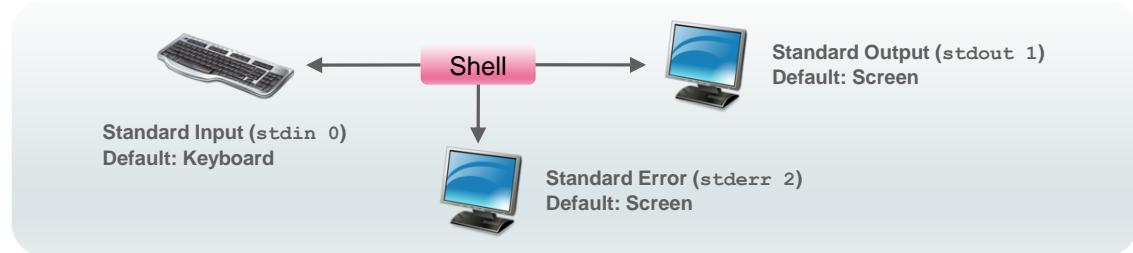


Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Caution:** Do not use these metacharacters when creating file and directory names. These symbols hold special meaning in the shell.

## Command Communication Channels

- By default, the shell receives or reads input from the standard input—the keyboard—and displays the output and error messages to the standard output—the screen.



- Input redirection forces a command to read the input from a file instead of from the keyboard.
- Output redirection sends the output from a command into a file instead of sending the output to the screen.

ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## File Descriptors

- Each process works with three file descriptors.
- File descriptors determine where the input to the command originates and where the output and error messages are directed to.
- The table explains the file descriptors.

| File Descriptor Number | File Description Abbreviation | Definition              |
|------------------------|-------------------------------|-------------------------|
| 0                      | stdin                         | Standard command input  |
| 1                      | stdout                        | Standard command output |
| 2                      | stderr                        | Standard command error  |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Redirection Metacharacters

Command redirection is enabled by the following shell metacharacters:

- Redirection of standard input (<)
- Redirection of standard output (>)
- Redirection of standard output (>>) append
- Redirection of standard error (2>)
- Redirection of both standard error and standard output (2>&1) to the same file
- The pipe symbol (|)



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Redirecting Standard Input (`stdin`)

- The less-than (<) metacharacter processes a file as the standard input instead of reading the input from the keyboard.

```
command < filename
or
command 0< filename
```

- Use the `dante` file as the input for the `mailx` command.

```
$ mailx oracle < ~/lab/dante
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Redirecting Standard Output (stdout)

- The greater-than (>) metacharacter directs the standard output to a file instead of printing the output to the screen.

```
command > filename
or
command 1> filename
and
command >> filename # appends
```

- If the file does not exist, the shell **creates** it. If the file exists, the redirection **overwrites** the content of the file, and the >> **appends** the output to the end of the file.
- Redirect the list of files and subdirectories of your current home directory into a **directory\_list file**.

```
$ cd
$ pwd
/home/oracle
$ ls -l > directory_list
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

When you use a single greater-than (>) metacharacter, the command overwrites the original contents of the file, if the file already exists. When you use two greater-than (>>) symbols, the command **appends** the output to the original content of the file.

```
$ command >> filename
```

Append the “That’s my directory\_list file” string to the end of the **my\_file** file.

```
$ ls -l > my_file; cat my_file
-rw-r--r-- 1 oracle oracle 1319 Jun 28 2009 dante
drwxr-xr-x 5 oracle oracle 512 Jun 28 2009 dir1
... (output truncated)
$ echo "That's my directory_list file" >> my_file; cat my_file
-rw-r--r-- 1 oracle oracle 1319 Jun 28 2009 dante
drwxr-xr-x 5 oracle oracle 512 Jun 28 2009 dir1
... (output truncated)
That's my directory list file
```

**Note:** The semicolon (;) is a shell metacharacter that allows you to use multiple commands on a single command line.

## Redirecting Standard Error (stderr)

- A command using the file descriptor number (2) and the greater-than (>) sign redirects any standard error messages to the /dev/null file (delete them).

```
command 2>/dev/null
```
- The following example shows the standard output and the standard error redirected to the dat file.

```
$ ls /var /test 1> dat 2>&1
$ less dat
ls: cannot access /test: No such file or directory (stderr)
/var: (stdout)
adm (stdout)
... (output truncated)
```

**Note:** The syntax 2>&1 instructs the shell to redirect stderr (2) to the same file that receives stdout (1).



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Pipe Symbol

- The pipe ( | ) metacharacter redirects the standard output from one command to the standard input of another command.
- The first command writes the output to standard output and the second command reads standard output from the previous command as standard input.

```
command1 | command2
```

- Use the standard output from the `who` command as the standard input for the `wc -l` command.

```
$ who | wc -l
35
```

**Note:** You can use pipes to connect several commands.

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The output of the `who` command never appears on the terminal screen because it is piped directly into the `wc -l` command.

## Using the Pipe Symbol

- To view a list of all the subdirectories located in the /etc directory, enter the following command.

```
$ ls -F /etc | grep "/"
X11/
acct/
apache/
apache2/
apoc/
... (output truncated)
```

- Use the output of the head command as the input for the tail command and print (lp - line printer) the results.

```
$ head -10 dante | tail -3 | lp
request id is printerA-177 (Standard input)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Redirecting Standard Output (stdout) by Using the tee Command

- As you saw earlier, the greater-than (>) metacharacter directs the standard output to a file instead of printing the output to the screen.  
`command > filename`
- Hypothetically, if you wanted to see the output from command1 before it is redirected to a file name, you would use the `tee` command.  
`command1 | tee [-a] filename`
- When using the `tee` command, if the file does not exist, the shell **creates** it. If the file exists, the `tee` redirection **overwrites** the contents of the file, and if the `[-a]` option (**append**) is used, the redirected output is appended to the end of the file.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Quoting Symbols

- Quoting is a process that instructs the shell to mask or ignore the special meaning of shell metacharacters.
- The quoting symbols are:
  - Apostrophe or single forward quotation marks (' '): Instructs the shell to ignore all enclosed metacharacters
  - Double quotation marks (" "): Instructs the shell to ignore all enclosed metacharacters and white space, except for the following three symbols:
    - Backslash (\) “escape symbol”: Prevents the shell from interpreting the next symbol after the (\) as a metacharacter
    - Single backward quotation marks (` `) backquote or backtick: Instructs the shell to execute and display the output for a command enclosed within the backward quotation marks
    - Dollar sign and parentheses \$(command): Instruct the shell to execute and display the output of the command enclosed within the parentheses



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To ignore the special meaning of the dollar (\$) metacharacter, enter the following command:

```
$ echo '$SHELL'
$SHELL
$ echo $SHELL
/bin/bash
```

Observe that the `echo` utility writes arguments to standard output.

## Quiz



The `ls -l 2> directory_list` command lists the content of your current directory and redirects that list into a file called `directory_list`.

- a. True
- b. False



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Practice 5: Overview

This practice covers only the highlighted topics:

- 5-1: Using Shell Metacharacters
- 5-2: Using Command Redirection
- 5-3: Using Variables in the `bash` shell
- 5-4: Displaying Command History
- 5-5: Customizing the User's Work Environment



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 5 in your Activity Guide.

## Lesson Agenda

- Using Shell Expansion for Generating Shell Tokens
- Using Shell Metacharacters for Command Redirection
- **Using Variables in the `bash` Shell to Store Values**
- Displaying the Command History
- Customizing the User's Work Environment



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Variables: Introduction

- A variable/parameter is a temporary storage area in memory, which is either set by the user, shell, system, or any program that loads another program.
- There are two categories of variables:
  - Local shell variables apply only to the current instance of the shell and are used to set short-term working conditions.
  - Global environment shell variables are local shell variables that have been `export(ed)`. The `export(ed)` variables are a subset of the total shell variables and are valid for the duration of any `fork(ed)` or `spawn(ed)` subordinate session.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** When a shell variable name follows the dollar sign \$ symbol, the shell interprets that the value stored inside that variable is to be substituted (parameter expansion) at that point.

## Displaying Local Shell Variables

- The `echo` command displays the value stored inside a local shell variable using parameter expansion.
- The `set` command lists all local shell variables and their values.

```
$ echo $SHELL
/bin/bash
```

```
$ set
DISPLAY=:0.0
EDITOR=/bin/vim
ERRNO=13
FCEDIT=/bin/vim
HELPPATH=/usr/openwin/lib/locale:/usr/openwin/lib/help
HOME=/home/oracle
HZ=100
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** There can be hundreds of local shell variables.

## Displaying Global Environment Shell Variables

- The `echo` command displays the value stored inside an environment shell variable.

```
$ echo $SHELL
/usr/bin/bash
```

- The `env` command lists all global environment shell variables and their values.

```
$ env
SHELL=/usr/bin/bash
UID=1000
HOME=/home/oracle
USERNAME=oracle
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Setting and Unsetting Shell Variables

- To create a bash local shell variable.

```
$ history=50
$ echo $history
```

- To unset a local shell variable.

```
$ history=
$ echo $history
```

- To create a bash environment shell variable, use the `export` command.

```
$ export history=75
$ env | grep history
history=75
$ echo $history
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The `set` command is used to display the local shell variables and their values, and the `env` command is used to display the environment shell variables.

A variable is set and a value is assigned with the following syntax:

`var=value`

or

`VAR=value` (most of the hundreds of bash shell local variables are in uppercase.)

There is no space on either side of the equals (=) sign.

```
$ private=/home/oracle/lab/private
$ set | grep private
private=/home/oracle/lab/private
$ cd $private; pwd
/home/oracle/lab/private
```

## Default Bash Shell Variables

| Variable       | Meaning                                                                                                                       |
|----------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>EDITOR</b>  | Defines the default editor for the shell                                                                                      |
| <b>FCEDIT</b>  | Defines the editor for the <code>fc</code> command. Used with the history mechanism for editing previously executed commands. |
| <b>HOME</b>    | Sets the directory to which the <code>cd</code> command changes when no argument is supplied on the command line              |
| <b>LOGNAME</b> | Sets the login name of the user                                                                                               |
| <b>PATH</b>    | Specifies a colon-separated list of directories to be searched when the shell needs to find a command to be executed          |
| <b>PS1</b>     | Specifies the primary bash shell prompt: \$                                                                                   |
| <b>PS2</b>     | Specifies the secondary bash command prompt, normally: >                                                                      |
| <b>SHELL</b>   | Specifies the name of the shell (that is, <code>/usr/bin/bash</code> )                                                        |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table describes variables that are assigned default values by the bash shell on login. In addition to these variables there are parameters that contain shell scripting syntax.

## Customizing bash Shell Variables: PS1

- The shell prompt string is stored in the shell variable `PS1`, and you can customize it according to your preference.

```
$ PS1='[\u@\h \w]\$'
[oracle@ol7-server1 ~]$
```

- In this Oracle Linux example, the prompt displays the user's login name "`\u`", the system's host name "`\h`", and the current working directory "`\w`".
- This shell prompt displays the correct information even when the user logs in to different hosts.
- In Oracle Solaris the `PS1` variable is slightly different, observe the colon ":" between "`\h`" and "`\w`".

```
$ PS1='[\u@\h:\w]\$'
[oracle@s11-server1:~]$
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Customizing Shell Variables: PATH

- The `PATH` variable contains a list of directory path names, separated by colons.
- When executing a command on the command line, the shell searches the directories listed in the `PATH` variable from left to right, in sequence to locate that command.
- If the shell does not find the command in the list of directories, it displays a “not found” error message.
- To ensure that commands operate smoothly, you must include the respective directory in the `PATH` variable.
- The example in the notes pages illustrates the inclusion of the `/home/oracle/lab` directory into the `PATH` variable and the use of `which`, a bash shell built-in.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Include the `/home/oracle/lab` directory in the `PATH` variable and use `which` to show the directory in the `PATH` variable where the executable file is stored.

```
$ echo $PATH
/usr/bin:/usr/sbin
$ ls -l lab/hello.sh
-rwx----- 1 oracle oracle ... lab/hello.sh
$ which hello.sh
no hello.sh in /usr/bin /usr/sbin
$ PATH=$PATH:~/lab
$ echo $PATH
/usr/bin:/usr/sbin:/home/oracle/lab
$ which hello.sh
/home/oracle/lab/hello.sh
```

The `PATH` variable automatically passes the value to the subshells.

**Note:** The command output may vary from UNIX to Linux.

## Quiz



The `set` command lists all local shell variables and their values.

- a. True
- b. False



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Lesson Agenda

- Using Shell Expansion for Generating Shell Tokens
- Using Shell Metacharacters for Command Redirection
- Using Variables in the `bash` Shell to Store Values
- **Displaying the Command History**
- Customizing the user's Work Environment



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Introducing Command History

- The shell keeps a history of previously entered commands.
- There are two global shell variables `HISTFILESIZE` and `HISTSIZE` that control the number of history entries.
- This history mechanism enables you to view, repeat, or modify previously executed commands.
- By default, the `history` command displays all history entries to standard output.

```
$ history
...
109 date
110 cd /etc
111 touch dat1 dat2
112 ps -ef
113 history
```

**Note:** The output may vary based on the commands recorded in the `~/.bash_history` file when you exit a terminal session.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Displaying Previously Executed Commands

To display the last four commands.

```
$ history 4
111 touch dat1 dat2
112 ps -ef
113 history
114 history 4
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Use the ! Command to Re-execute a Command Line from History

- The exclamation symbol (!) command, also called “bang”, is an alias built-in to the bash shell, which enables you to repeat a command.
- The output from the history command shows a line number in front of the command line. Use “!###” to re-execute any command or use a relative location number, for example, “!-n”.

```
$ history
... (output truncated)
109 date
110 cd /etc
111 touch dat1 dat2
112 ps -ef
113 history
```

- Re-execute 112 ps -ef by using the ! command or by using relative positioning.

```
$!112
or
$!-2
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To re-execute the command cat dante using either the ! command or relative positioning:

```
$ history
...
122 cat dante
123 ls
124 cd ~/dir1
$!122
or
$!-3
```

## Use the !! Command to Repeat the Last Command

- The !! command is an alias built in to the bash shell, which enables you to repeat the last command.
- Repeat/re-execute the cal command by using “!!” or simply recall the last command by pressing the up arrow key and then press Return/Enter to execute.

```
$ cal
March 2017
Su Mo Tu We Th Fr Sa
 1 2 3 4
 5 6 7 8 9 10 11
12 13 14 15 16 17 18
... (output truncated)
$!!
cal
March 2017
Su Mo Tu We Th Fr Sa
 1 2 3 4
 5 6 7 8 9 10 11
12 13 14 15 16 17 18
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Searching the History Entries

- Pressing the Ctrl + R keys together initiates a search and prompts for the search string.

```
$ Ctrl+r
(reverse-i-search) ``:
```

- When entering the search string, bash returns the first found match from the bottom of the current working set of commands combined with the `~/.bash_history` file.
  - If you want to search for the last occurrence of the `clear` command, entering “`cl`” returns “`clear`”. Press Return/Enter to execute, or press Ctrl + C to cancel.

```
$ Ctrl+r
(reverse-i-search)`cl': clear
```

- If that is not the command you are looking for, pressing Ctrl + R again continues the reverse search.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The search for the string entered is case-sensitive as are most things in UNIX and Linux.

## Using the ! Command to Search for and Execute History Entries

You can search for **and** execute using “!” combined with a search string, for example “!cl”.

```
$!cl
```

**Caution:** Please ensure the string you entered is not a string in a destructive command; you may not get the results you are expecting.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To repeat the command containing the string dante.

```
$ history
...
122 cat dante
123 ls
124 cd ~/dir1
$!dante
```

## Editing Commands on the Command Line

- You can edit commands by using a shell inline editor.
- The default command-line editing mode in bash is emacs.
- You can, however, switch to vim (vi) mode as well.
- The set -o command switches between the two modes.

```
$ set -o vi
$ set -o emacs
```

- You can also set the editing mode by using the EDITOR or VISUAL shell variables.

```
$ export EDITOR=/bin/vim
or
$ export VISUAL=/bin/vim
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To access a command from the history buffer, edit the command with the vi editor, and execute the modified command by performing these steps:

1. Set the editing mode to vi.

```
$ set -o vi
vi on
```

2. Enter the history command to view the command history.

```
$ history
```

3. Now, use the vim commands to edit any previously executed command. Press the Esc key and use the following keys to move the cursor through the command history list.

- k: Moves the cursor up one line at a time
- j: Moves the cursor down one line at a time

After the desired line/command is located:

- l: Moves the cursor to the right
- h: Moves the cursor to the left

**Note:** In bash, you can use the arrow keys on the command line in both emacs and vi modes.

4. To run the modified command, press the Return/Enter key

## Invoking File Name Completion

- File name completion is a feature that allows you to enter the first part of a file name or directory name and press a key to fill out or complete the file name/directory name.
- To invoke file name completion, enter the desired command followed by one or more characters of a file name and then press the Esc/Tab keys or just press the Tab key.
- Expand a file name beginning with the letters `sb` in the `/usr` directory:

```
$ cd /usr
$ ls sb "Press the Tab key"
```

- The shell completes the remainder of the file name by displaying, `ls sbin/`.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## File Name Completion with More Than a Single Solution

- You can request the shell to present all possible alternatives of a partial file name from which you can select.
- This request can be invoked by pressing the Escape (Esc) and the equal (=) sign keys in sequence or by pressing Tab twice.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To request the shell, present all file names beginning with the letter “g” in the /etc directory, enter the following commands:

```
$ cd /etc
$ cat g "Press Esc followed by the = key" or press the Tab twice
gconf/ gnome-vfs-2.0/ group gtk-2.0/
getty gnome-vfs-mime-magic grock gtk/
gimp/ gnopernicus-1.0/ gss/
```

**Note:** Just like the ls -F command, files listed followed by a “/” are not files but directories.

The cursor is positioned after the letter “g.” At this point, continue typing with the next character to refine the search.

```
$ cat g_
```

If there are too many possibilities, the command completion displays a warning message to let you decide if you want to see all of the possibilities.

```
Display all 142 possibilities? (y or n)
```

## Lesson Agenda

- Using Shell Expansion for Generating Shell Tokens
- Using Shell Metacharacters for Command Redirection
- Using Variables in the `bash` Shell to Store Values
- Displaying the Command History
- Customizing the User's Work Environment



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## User Initialization Files

- Other than having a home directory to create and store files, users need an environment that gives them access to the tools and resources.
- When a user logs in to a system, the user's work environment is determined by the initialization files.
- These initialization files are defined by the user's startup shell, which can vary depending on the update or release.
- The default initialization files in your home directory enable you to customize your working environment.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Default User Initialization Files for the bash Shell

- When bash is invoked, it first reads and executes commands from the /etc/profile file, if the file exists.
- bash then reads and executes commands from the ~/.bash\_profile, ~/.bash\_login and ~/.bashrc, which executes /etc/bashrc, if it exists.
- In the absence of the aforementioned files, the ~/.profile file is executed.
- When a login shell exits, bash reads and executes commands from the ~/.bash\_logout file, if it exists.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

- The /etc/profile file is a systemwide file that the system administrator maintains. This file defines tasks that the shell executes for every user who logs in. The instructions in the file usually set the shell variables, such as PATH, USER, and HOSTNAME.
- The ~/.bash\_profile file is a configuration file for configuring user environments. The users can modify the default settings and add any extra configurations in it.
- The ~/.bash\_login file contains specific settings that are executed when a user logs in to the system.
- The ~/.bashrc file contains specific settings that are executed when a user logs in to the system.
- The /etc/bashrc file contains specific settings that are executed when a user logs in to the system.
- The ~/.profile file is yet another configuration file that is read in the absence of the ~/.bash\_profile and ~/.bash\_login files.
- The ~/.bash\_logout file contains instructions for the logout procedure.

## Configuring the `~/.bashrc` File

- The `~/.bashrc` file is a personal initialization file for configuring the user environment.
- The file is defined in your home directory and can be used for the following:
  - Modifying your working environment by setting custom shell environment variables and terminal settings
  - Instructing the system to initiate applications
- However, before the changes can be instantiated the `~/.bashrc` file has to be reread.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

In Oracle Linux, use the `cat` command to display the contents of the `.bashrc` file.

```
$ cat -n .bashrc
1 # .bashrc
2
3 # Source global definitions
4 If [-f /etc/bashrc]; then
5 . /etc/bashrc
6 fi
7
8 # Uncomment the following line if you don't like systemctl's auto-paging
feature:
9 # export SYSTEMD_PAGER=
10
11 # user specific aliased and functions
```

In Oracle Solaris, use the `cat` command to display the contents of the `.bashrc` file.

```
$ cat -n .bashrc
1 #
2 # Define default prompt to <username>@<hostname>:<path><" ($|#) "'>
3 # and print '#' for user "root" and '$' for normal users.
4 #
5
6
7 typeset +x PS1="\u@\h:\w]\$\\"
```

## Rereading the `~/.bashrc` File

- There are two ways to reread the `~/.bashrc`:
  - exit the current terminal session and restart a new terminal session.
  - Use a bash shell built-in called `source` also aliased as `(.)` a period to reread the `~/.bashrc` file in the current shell without `fork(ing)` or `spawn(ing)` a subordinate shell.

```
$ source ~/.bashrc
or
$. ~/.bashrc
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Quiz



Which of the following is the default command-line editing mode in bash?

- a. vi
- b. ed
- c. emacs
- d. vim



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Summary

In this lesson, you should have learned how to:

- Use shell expansion for generating shell tokens
- Use shell metacharacters for command redirection
- Use variables in the `bash` shell to store values
- Display the command history
- Customize the user's work environment



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Practice 5: Overview

This practice covers only the highlighted topics:

- 5-1: Using Shell Metacharacters
- 5-2: Using Command Redirection
- 5-3: Using Variables in the `bash` shell
- 5-4: Displaying Command History
- 5-5: Customizing the User's Work Environment



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 5 in your Activity Guide.

# Using Basic File Permissions

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Objectives

After completing this lesson, you should be able to:

- View file and directory permissions
- Change ownership
- Change permissions
- Modify default permissions



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Lesson Agenda

- Viewing File and Directory Permissions
- Changing Ownership
- Changing Permissions
- Modifying Default Permissions



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Securing Files and Directories

- One of the important functions of a secure system is to limit access to authorized users and prevent unauthorized users from accessing the files or directories.
- UNIX and Linux use two basic means to prevent unauthorized access to a system:
  - To authenticate both a privileged user account and an unprivileged user account by verifying that the username and password exist and have been correctly entered
  - To protect file and directory access, the UNIX and Linux OSes assign a standard set of access permissions at the time of file and directory creation. These permissions are called an Access Control List (ACL).



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The UNIX and Linux OSes also provide a special user account on every system, called the `root` user/role. The `root` user/role, often referred to as the superuser, has complete access to every user account and all files and directories. The `root` user/role can override the ACL permissions placed on all files and directories.

## File and Directory Permissions (ACL)

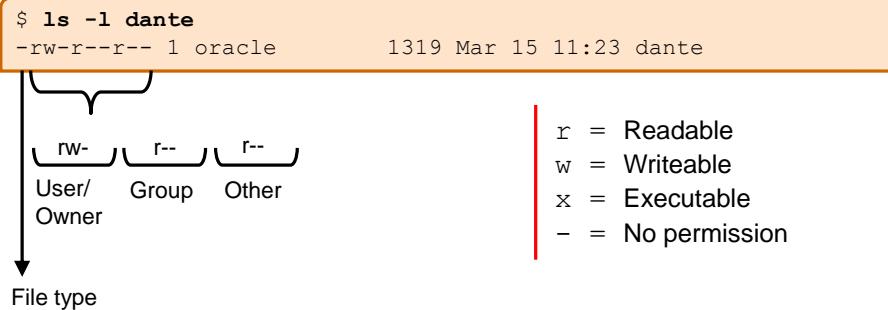
- All files and directories in UNIX and Linux have a default set of standard access permissions.
- These access permissions control who can access what files, and provides a fundamental level of security to the files and directories in a system.
- The standard set of access permissions are established by a user's `umask` settings. The `umask` command is described in more detail later in this lesson.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Viewing Permission Categories

To view the permissions for files and directories, use the `ls -l` or `ls -n` commands.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

- The first field of information displayed by the `ls -l` command is the file type. The file type typically specifies whether it is a file or a directory. A file is represented by a hyphen (-). A directory is represented by the letter `d`.
- The remaining fields represent the permission groups: user/owner, group, and other.

## Permission Groups

- There are three permission groups:
  - User (Owner) who owns the file or directory
  - Group
  - Other
- The table describes the permission groups and their scope:

| Permission Groups | Description                                                                                                                  |
|-------------------|------------------------------------------------------------------------------------------------------------------------------|
| User/Owner (u)    | Permissions used by the assigned user/owner of the file or directory                                                         |
| Group (g)         | Permissions used by members of the group that owns the file or directory                                                     |
| Other (o)         | Permissions used by <b>all</b> users other than the file owner, and members of the group that owns the file or the directory |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Permission Sets

- Each permission group has three permissions, called a permission set.
- Each set consists of **read**, **write**, and **execute** permissions that are represented by the characters **r**, **w**, and **x**, respectively.
- Each file or directory has three permission sets for the three types of permission groups.
- The first permission set represents the user/owner permissions, the second set represents the group permissions, and the last set represents the other (world) permissions.
- The presence of any of these characters, such as **r**, indicates that the particular permission is granted.
- A dash (-) symbol in place of a character in a permission set indicates that a particular permission is denied.
- UNIX and Linux assign initial permissions automatically based on a user's **umask** when a new file or directory is created.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The system administrator creates and maintains groups in the `/etc/group` file. The system administrator assigns users to groups according to the need for shared file access.

## Interpreting File and Directory Permissions

| Permissions        | Access for a File                                                                                                                            | Access for a Directory                                                                                                                                                                                                                                                                                    |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Read (r)</b>    | You can display file contents and copy the file.                                                                                             | You can list the directory contents with the <code>ls</code> command.                                                                                                                                                                                                                                     |
| <b>Write (w)</b>   | You can modify the file contents, but only if you also have <b>read</b> permissions.                                                         | You can modify the contents of a directory, by deleting ( <code>rm</code> ) a file. You must also have the <b>execute</b> permission for this to happen.                                                                                                                                                  |
| <b>Execute (x)</b> | You can execute the file if it is an executable. You can execute a shell script if you also have <b>read</b> and <b>execute</b> permissions. | You can use the <code>cd</code> command to access the directory. If you also have <b>read</b> access, you can run the <code>ls -l</code> command on the directory to list the contents. If you do not have <b>read</b> access, you can run the <code>ls</code> command as long as you know the file name. |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The read, write, and execute permissions are interpreted differently when assigned to a file than when assigned to a directory. The table in the slide shows the permission definitions for a file and directory.

**Note:** For a directory to be of general use, it must at least have read and execute permissions.

## Determining File or Directory Access Permissions

- The `ls -l` and `ls -n` commands display the ownership of files and directories and their corresponding permissions.
- All files and directories have an associated `username` and a user identification number (UID) and a `group name` and a group identification number (GID).
- To view the UIDs and GIDs, run the `ls -n` command on the `/var/adm` directory.

```
$ ls -n /var/adm
total 244
drwxrwxr-x 5 4 4 512 Nov 15 14:55 acct
-rw----- 1 5 2 0 Jun 7 12:28 aculog
drwxr-xr-x 2 4 4 512 Jun 7 12:28 exacct
-r-----r-- 1 0 0 308056 Nov 19 14:35 lastlog
drwxr-xr-x 2 4 4 512 Jun 7 12:28 log
... (output truncated)
```

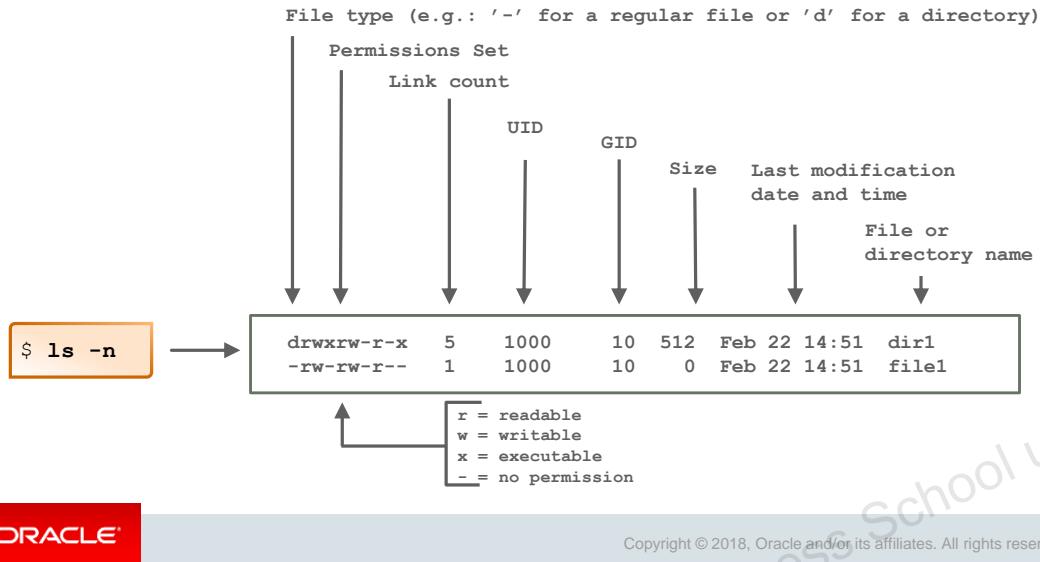


Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The UID identifies the user who owns the file or directory. The GID identifies the group of users who own the file or directory. A file or directory can belong to only one group at a time. The UNIX and Linux OSes use these numbers to track ownership and group membership of files and directories.

## Interpreting the ls -n Command

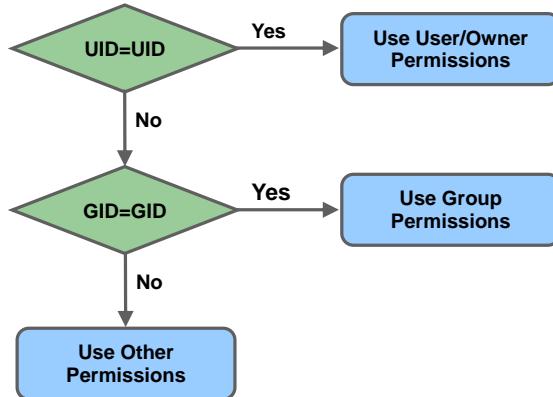
- The ls -n command displays the UID and GID listing of file information.



The image in the slide illustrates the parts of the output of the `ls -n` command.

- The first character is the file/directory type.
- The next nine characters are the three permission sets.
- The next character represents the count of hard links to the file or directory. A hard link is a pointer that shows the number of files or directories a particular file is linked to within the same file system.
- The next set of characters represents the UID of the user/owner (1000).
- The next set of characters represents the GID of the group (10).
- The next set of characters represents the size of the file or directory in bytes.
- The next set of characters represents the time and date the file or directory was last modified.
- The last set of characters represents the name of the file or directory.

## Determining Permissions



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

When a user attempts to access a file or directory, the UID of the user is compared with the UID of the file or directory. If the UIDs match, the permission set for the owner determines whether the owner has access to the file or directory.

If the UIDs do not match, the user's GID is compared with the GID of the file or directory. If these numbers match, the group permissions apply.

If the GIDs do not match, the permission set for other is used to determine file and directory access.

The image in the slide shows the decision tree for determining file and directory permissions.

- If the UID equals the UID, then use the user/owner permissions.
- If not, does the GID equal the GID? If yes, use group permissions. If not, use other permissions.

## Quiz



Which of the following directories have read and execute permissions set for the owner and group only?

- a. dr-xr-x---
- b. dr-x---r-x
- c. d---r-xr-x



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Lesson Agenda

- Viewing File and Directory Permissions
- **Changing Ownership**
- Changing Permissions
- Modifying Default Permissions



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Changing Ownership on Files or Directories

- Every file and directory in UNIX and Linux is owned by somebody.
- The `ls -l` command shows the `username` and the `group` that owns the object.
- The `ls -n` command shows the `UID` and `GID` numbers corresponding to who owns the object.
- There are two commands:
  - The `chown` command can be used to change both `username` and `group` ownership.
  - The `chgrp` command changes only `group` ownership.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Changing Both username and group Ownership

- The syntax for the `chown` command is:

```
$ chown [options] [newusername] [:newgroup] filename
```

```
$ ls -l dante
-rw-r--r-- 1 student class 1319 Mar 15 11:23 dante
$ chown oracle:oracle dante
$ ls -l dante
-rw-r--r-- 1 oracle oracle 1319 Mar 15 11:23 dante
```

- You can change the ownership only for files and directories that you own. However, the system administrator can change the ownership of any object.
- For more information about the `chown` command options, see the `chown` man pages.

**Caution:** If you change the `username` ownership of a file or directory, you have just given that object away, and you cannot get it back without help from the system administrator.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Changing group Ownership

- The syntax for the `chgrp` command is:

```
$ chgrp [options] newgroup filename
```

- For more information about the `chgrp` command options, see the `chgrp` man pages.

**Note:** If you still own a file or directory, you can always change the group ownership.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

To change group membership on files and directories:

```
$ ls -l dante
-rw-r--r-- 1 oracle class 1319 Jan 22 14:51 dante
$ chgrp oracle dante
$ ls -l dante
-rw-r--r-- 1 oracle oracle 1319 Jan 22 14:51 dante
```

## Quiz



You can give away a file or directory ownership that you cannot get back.

- a. True
- b. False



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Lesson Agenda

- Viewing File and Directory Permissions
- Changing Ownership
- **Changing Permissions**
- Modifying Default Permissions



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Changing Permissions

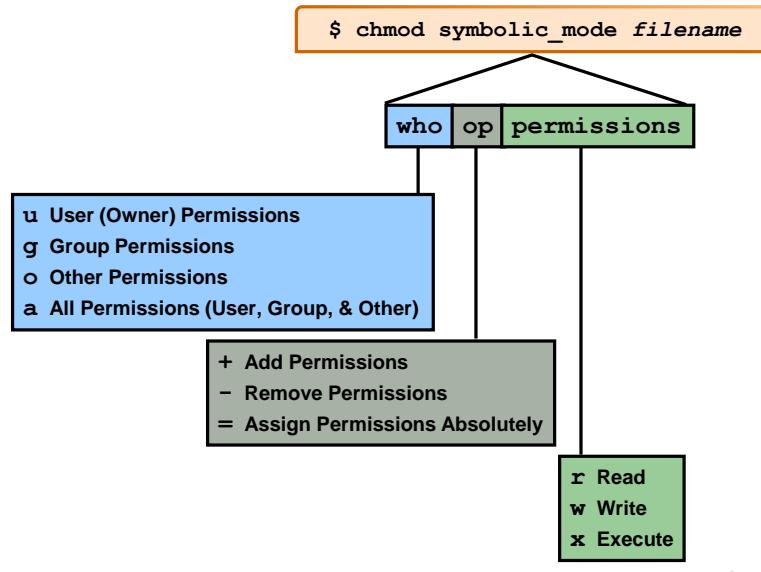
- You can change the permissions on files and directories by using the `chmod` command.
- Either the user/owner of the file or directory, or the `root` user can use the `chmod` command to change permissions.
- The `chmod` command can be used in either symbolic or octal mode.
  - **Symbolic mode** uses a combination of letters and symbols to add or remove permissions for each permission group.
  - **Octal mode**, also called absolute mode, uses octal numbers to represent each permission group.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You can assign execute permissions on files with the `chmod` command. The `chmod` command is described later in this lesson. Execute permissions are not assigned by default when you create a file.

## Changing Permissions: Symbolic Mode



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The image in the slide shows the components of the symbolic mode command syntax. The first four letters represent the “who” and consist of the following codes:

- u: User (Owner) permission
- g: Group permissions
- o: Other permissions
- a: All permissions (user, group, and other)

The next section is the “op” section and consists of the following:

- +: Add permissions
- -: Remove permissions
- =: Set permissions equal

The last section is the “permissions” section and consists of the following:

- r: Read
- w: Write
- x: Execute

## Changing Permissions: Symbolic Mode

- The syntax for the `chmod` command in symbolic mode is:

```
$ chmod [options] [symbolic_mode] filename
```

- The format of the `symbolic_mode` consists of three parts: `[ugoa] [+-=] [rwx]`
  - The user category `[ugoa]`: User/owner, group, other, or all
  - The function to be performed `[+=]`: Add, remove or set equal
  - The permissions affected `[rwx]`: Read, write, and execute  
Plus special file permissions and *sticky bit* `[st]` (described in the next slide)
- If the option is `g+x`, the executable permission is added to the *group* permissions.
- For more information about the `chmod` command options, see the `chmod` man pages.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following examples illustrate how to modify permissions on files and directories by using symbolic mode. To remove the read permission for other users, run the following commands:

```
$ ls -l dante
-rw-r--r-- 1 oracle oracle 1319 Jan 22 14:51 dante
$ chmod o-r dante
$ ls -l dante
-rw-r---- 1 oracle oracle 1319 Jan 22 14:51 dante
```

To remove the read permission for the group, run the following commands:

```
$ chmod g-r dante
$ ls -l dante
-rw----- 1 oracle oracle 1319 Jan 22 14:51 dante
```

## Special File Permissions: Setuid, Setgid, and Stick Bit

- When special file permissions are set on executable files, the user who runs that executable file executes it with the permissions of the UID or GID, who has the “s” in place of the “x” for executable privileges.

```
$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 27856 May 4 2014 /usr/bin/passwd
```

- The restricted deletion flag “t” (sticky bit) is a permission bit that protects the files within a directory from being deleted by anyone, except the user who owns the file, the owner of the directory, or the root user.

```
$ ls -l /tmp
drwxrwxrwt 68 root root 4096 Apr 15 07:02 /tmp
```

- Typically, only the /tmp directory has the “t” in place of the “x”, which makes it “world readable and world writable”.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You can also use the find command with expressions to identify files with special file permissions. The following find command with the -perm 4000 option searches for the user/owner permissions that has an “s” in place of the “x” permission.

```
$ find /usr/bin -perm -4000 -exec ls -l {} \;
...
-rwsr-xr-x. 1 root root 27856 May 4 2016 /usr/bin/passwd
-rwsr-xr-x. 1 root root 57536 Jul 28 2016 /usr/bin/crontab
---s--x--x. 1 root root 130712 Dec 13 2016 /usr/bin/sudo
```

The following find command with the -perm 2000 option searches for the group permissions that has an “s” in place of the “x” permission.

```
$ find /usr/bin -perm -2000 -exec ls -l {} \;
...
-rwx--s--x. 1 root slocate 40536 May 3 2016 /usr/bin/locate
```

The following find command with the -type d (directory) and -perm -1000 options searches for the other permissions that has a “t” in place of the “x” permission.

```
$ find / -type d -perm -1000 -exec ls -ld {} \;
...
drwxrwxrwt. 14 root root 4096 May 14 2017 /tmp
```

## Changing Permissions: Octal Mode

- The syntax for the `chmod` command in octal mode is:

```
$ chmod [octal_mode] filename
```

- The `octal_mode`, sometime called the `absolute_mode`, option consists of three octal numbers, 4, 2, and 1, that represent a combination (sum) of the permissions, from 0–7, for the file or directory.

| Octal Value | Permission |
|-------------|------------|
| 4           | Read       |
| 2           | Write      |
| 1           | Execute    |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide shows the octal numbers for each individual permission. These numbers are combined into one number for each permission set.

## Changing Permissions: Octal Mode

| Octal Value | Permission               | Binary      |
|-------------|--------------------------|-------------|
| 7           | rwx                      | 111 (4+2+1) |
| 6           | rw-                      | 110 (4+2+0) |
| 5           | r-x                      | 101 (4+0+1) |
| 4           | r--                      | 100 (4+0+0) |
| 3           | -wx (see the notes page) | 011 (0+2+1) |
| 2           | -w- (see the notes page) | 010 (0+2+0) |
| 1           | --x (see the notes page) | 001 (0+0+1) |
| 0           | ---                      | 000 (0+0+0) |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide shows the octal numbers that represent a combined (sum) of a set of permissions.

**Note:** The write (`w`) and execute (`x`) permissions also require the read (`r`) permission. Therefore, you cannot write or execute a file if you cannot read it.

Therefore, the permission sets: `-wx`, `-w-`, and `--x` are **not** useful.

## Changing Permissions: Octal Mode

- You can modify the permissions for each category of users by combining the octal numbers.
- The first set of octal numbers defines user/owner permissions, the second set defines group permissions, and the third set defines other permissions.

| Octal Mode | Permissions                    |
|------------|--------------------------------|
| 640        | rw-r-----                      |
| 644        | rw-r--r--                      |
| 750        | rwxr-x---                      |
| 751        | rwxr-x--x (see the notes page) |
| 755        | rwxr-xr-x                      |
| 777        | rwxrwxrwx                      |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide shows the permission sets in octal mode.

**Note:** The write (w) and execute (x) permissions also require the read (r) permission. Therefore, you cannot write or execute a file if you cannot read it!

So the permission set: rwxr-x--x is **not** useful.

## Changing Permissions: Octal Mode

- Set permissions so that the owner, group, and other have read and execute access only.

```
$ chmod 555 dante
$ ls -l dante
-rwxr-xr-x 1 oracle oracle 1319 Jan 22 14:51 dante
```

- The `chmod` command fills in any missing octal digits to the left with zeros.

```
$ chmod 44 dante
$ ls -l dante
----r--r-- 1 oracle oracle 1319 Jan 22 14:51 dante
```

**Note:** `chmod 44 dante` becomes `chmod 044 dante`.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Caution:** Not using the correct octal values or leaving one or more of the values missing can lead to unwanted access to files or directories. Some additional examples show how to modify permissions on files and directories by using octal mode.

Change owner and group permissions to include write access.

```
$ chmod 775 dante
$ ls -l dante
-rwxrwxr-x 1 oracle oracle 1319 Jan 22 14:51 dante
```

Change the group permissions to read and execute only.

```
$ chmod 755 dante
$ ls -l dante
-rwxr-xr-x 1 oracle oracle 1319 Jan 22 14:51 dante
```

## Quiz



What is the correct octal value for the “write and execute” file permission?

- a. 3
- b. 5
- c. 6
- d. 7



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Quiz



What is the correct permission set for the `rwxr-xr-x` octal mode?

- a. 775
- b. 644
- c. 755
- d. 674



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Lesson Agenda

- Viewing File and Directory Permissions
- Changing Ownership
- Changing Permissions
- Modifying Default Permissions



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Umask: A bash Shell Built-in Command

- When files and directories are created, initial permission values are automatically assigned.
- The initial maximum permission value for a file is 666 (`rw-rw-rw-`) and 777 (`rwxrwxrwx`) for a directory.
- The user mask affects and modifies the default file permissions assigned to the file or directory.
- You can set the user's mask by using the `umask` command in a user initialization file.
- To view the `umask` value, run the `umask` command.

```
$ umask
0022
```

**Note:** The default `umask` value for users in Oracle Solaris is 0022 (022), whereas the default value for Oracle Linux is 0002 (002).



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The `umask` utility affects the initial permissions for files and directories when the files and directories are created. The `umask` utility is a four-digit octal value that is associated with the read, write, and execute permissions.

- The first digit determines the Setuid or Setgid bit on the execute privileges.
- The second digit determines the default permissions for the user/owner.
- The third digit determines the default permissions for the group.
- The fourth digit determines the default permissions for other.

To set the default file permissions in a user initialization file to `rw-rw-rw-`, run the following command:

```
$ umask 0000
```

## Determining the `umask` Octal Value

| umask Octal Value | File Permissions         | Directory Permissions    |
|-------------------|--------------------------|--------------------------|
| 0                 | rw-                      | rwx                      |
| 1                 | rw-                      | rw-                      |
| 2                 | r--                      | r-x                      |
| 3                 | r--                      | r--                      |
| 4                 | -w- (see the notes page) | -wx (see the notes page) |
| 5                 | -w- (see the notes page) | -w- (see the notes page) |
| 6                 | ---                      | --x (see the notes page) |
| 7                 | ---                      | --- (none)               |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide shows the file and directory permissions for each of the `umask` octal values. This table can also help you determine the `umask` value that you want to set on files and directories. To determine the `umask` value, subtract the value of the permissions that you want from 666 for a file, or 777 from a directory.

If you want to change the default mode for files to 644 (`rw-r--r--`). The difference between 666 and 644 is 0022, which is the value you would use as an argument to the `umask` command.

**Note:** The write (w) and execute (x) permissions also require the read (r) permission. Therefore, you cannot write or execute a file if you cannot read it.

Therefore, the permission sets: `-wx`, `-w-`, and `--x` are **not** useful.

## Applying the umask Value

- When you mask out certain permissions from the initial value, the default permissions assigned to the new files and directories remain.
- The table displays the results in symbolic mode.

| Permission Field        | Description                                               |
|-------------------------|-----------------------------------------------------------|
| <code>-rw-rw-rw-</code> | Initial value specified by the system for a new file      |
| <code>w w</code>        | Default umask utility value to be removed (0022)          |
| <code>-rw-r--r--</code> | Default permissions assigned to newly created files       |
| <code>drwxrwxrwx</code> | Initial value specified by the system for a new directory |
| <code>w w</code>        | Default umask utility value to be removed (0022)          |
| <code>drwxr-xr-x</code> | Default permissions set for newly created directories     |



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The initial permission for a new file in symbolic mode is `rw-rw-rw-`. This set of permissions corresponds to read/write access for the owner, group, and other. This value is represented in octal mode as 420420420 or 666.

- To mask out the write permission for group and other, use 0022, the default umask value.
- The result is 420400400 or 644 in octal mode, and `rw-r--r--` in symbolic mode.

You can apply this same process to determine the default permissions for directories.

For directories, the initial value specified by the system is `rwxrwxrwx`. This corresponds to read, write, and execute access for user/owner, group, and other. This value is represented in octal mode as 421421421 or 777.

- Use the default umask value of 0022 to mask out the write permission for group and other.
- The result is 421401401 or 755 in octal mode, and `rwxr-xr-x` in the symbolic mode.

## Changing the umask Value

- You can change the `umask` value to a new value on the command line.
- For example, you might require a more secure `umask` value of say 027, which assigns the following access permissions to newly created files and directories:
  - Files with read and write permissions for the user/owner, read permission for the group, and no permissions for other (`rwxr--r---`), octal value 640
  - Directories with read, write, and execute permissions for the user/owner, read and execute permissions for the group, and no permissions for other (`rwxr-x---`), octal value 750

```
$ umask 027
$ umask
0027
```

**Note:** The default `umask` is set in `/etc/profile` file.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The new `umask` value affects only those files and directories that are created from this point onward. However, if the user logs out of the system, the new value (027) is replaced by the original value (022) on subsequent logins because the `umask` value was changed using the command line.

## Summary

In this lesson, you should have learned how to:

- View file and directory permissions
- Change ownership
- Change permissions
- Modify default permissions



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Practice 6: Overview

This practice covers the following topics:

- 6-1: Changing File Ownership
- 6-2: Changing File Permissions
- 6-3: Modifying Default Permissions



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 6 in your Activity Guide.

# Performing Basic Process Control

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Objectives

After completing this lesson, you should be able to:

- Describe a process and its attributes
- Manage processes



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Agenda

- Describing a Process and Its Attributes
- Managing Processes



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Process: Overview

- A process, also known as a task, is the running form of a program or a shell script.
- Programs and scripts are stored on disk and processes run in memory.
- Processes have a parent/child relationship.
- A running process can spawn one or more child processes, or `fork` multiple independent processes.
- Multiple processes can run in parallel.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Attributes of a Process

- The kernel assigns a unique identification number to each process called a process ID or PID.
  - The kernel uses this PID to track, control, and manage the process.
  - Every child PID has an owning Parent Process ID (PPID).
- Each process is further associated with a UID and a GID.
  - UIDs and GIDs indicate the process's owner.
  - Generally, the UID and GID associated with a process are the same as the UID and GID of the user who started the process.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

A process consists of an address space and a metadata object. The process space pertains to all the memory and swap space a process consumes. The process metadata is just an entry in the kernel's process table and stores all other information about a process.

## Process States

- The `s`, `stat`, and `state` output specifiers describe the state of a process.
- A process may be in any one of the following states:
  - `D`: Uninterruptible sleep (usually IO)
  - `R`: Running or runnable (on run queue)
  - `S`: Interruptible sleep (waiting for an event to complete)
  - `T`: Stopped, either by a job control signal or because it is being traced
  - `Z`: Defunct (“zombie”) process, terminated but not reaped by its parent
- For more information about the `ps` command process state options, see the `ps` man page.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

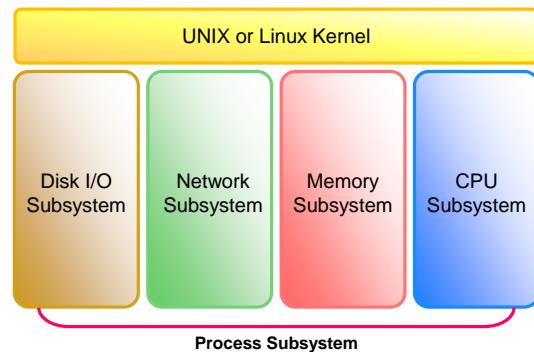
The process state can be displayed using the `ps` command. For **UNIX**, **BSD**, and **GNU Long options** when the `stat` keyword is used, additional state information is displayed such as the following:

- `<`: High-priority (not nice to other users)
- `N`: Low-priority (nice to other users)
- `L`: Has pages locked into memory (for real time and custom IO)
- `s`: Is a session leader
- `t`: Is multi-threaded
- `+`: Is in the foreground process group

**Note:** `nice` is a useful program that is used to decrease or increase the scheduling priority of a process or batch processes. Users can assign `nice` values between -20 (most favorable), 0 (no effect) and 19 (least favorable). The higher the `nice` value, the lower the scheduling priority.

# Process Subsystems

- Each time you boot a system, execute a command, or start an application, the system activates one or more processes.
  - A process, as it runs, uses the resources of the various subsystems:
    - Disk I/O
    - Network
    - Memory
    - CPU



A process, as it runs, uses the resources of the various subsystems:

- **The disk I/O subsystem:** Controls disk utilization and resourcing as well as file system performance
  - **The network subsystem:** Controls the throughput and directional flow of data between systems over a network connection
  - **The memory subsystem:** Controls the utilization and allocation of physical, virtual, and shared memory
  - **The CPU subsystem:** Controls CPU resources, loading, and scheduling

If not monitored and controlled, processes can consume your system resources, causing the system to run slowly and, in some cases, even halt or crash. The UNIX or Linux kernel collects performance-relevant statistics on each of these subsystems, to include process information. You can view and use this information to assess the impact that the processes have on the subsystem resources.

## Agenda

- Describing a Process and Its Attributes
- Managing Processes



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## List System Processes

- The process status (`ps`) command lists the processes that are associated with your shell.

```
$ ps [options]
```
- For each process, the `ps` command displays the PID, the terminal identifier (TTY), the cumulative execution time (TIME), and the command name (CMD).
- List the currently running processes on the system owned by the logon user using the `ps` command.

```
$ ps
PID TTY TIME CMD
1001 pts/1 0:00 bash
1004 pts/1 0:00 ps
```
- For more information about the `ps` command options, see the `ps` man page.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

The `ps` command has several options that you can use to display additional process information.

- `-e`: Prints information about every process currently running
- `-f`: Generates a full listing
- `-l`: Generates a long listing
- `-o format`: Writes information according to the format specification given in a format. Multiple `-o` options can be specified. The format specification is interpreted as the space-character-separated concatenation of all the format option arguments.

## Listing All Processes

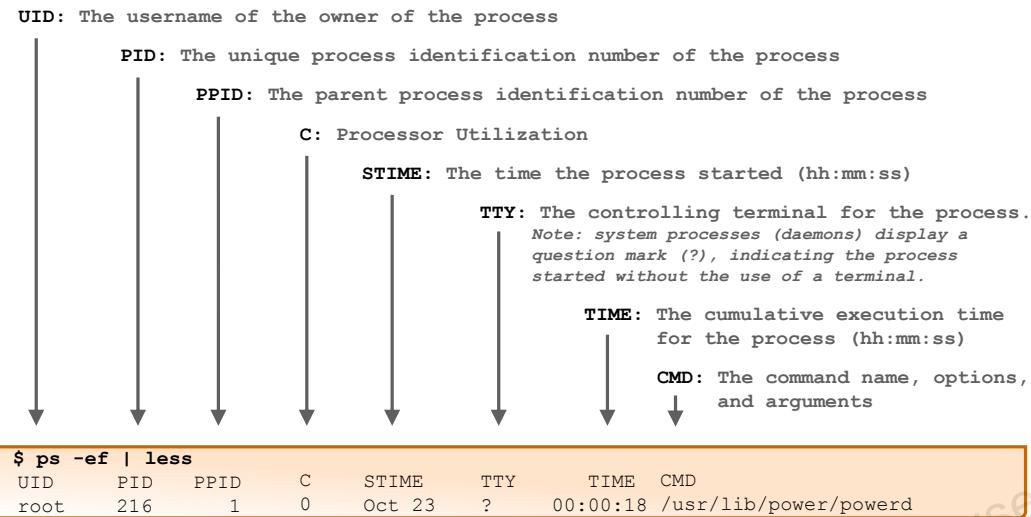
Use the `ps -ef` command to list the full-format of all the processes currently scheduled to run on the system.

```
$ ps -ef | less
UID PID PPID C STIME TTY TIME CMD
root 0 0 0 Feb 13 ? 00:00:18 sched
root 1 0 0 Feb 13 ? 00:00:01 /etc/init -
root 2 0 0 Feb 13 ? 00:00:00 pageout
root 3 0 0 Feb 13 ? 00:17:47 fsflush
root 9 1 0 Feb 13 ? 00:00:00 svc.configd
... (output truncated)
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Listing All Processes



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

The illustration in the slide interprets the output of the `ps -ef` command.

- The first column is the `UID`, the username of the owner of the process.
- The second column is the `PID`, the unique process identification number of the process.
- The third column is the `PPID`, the parent process identification number of the process.
- The fourth column is `C`, processor utilization
- The fifth column is the `STIME`, the time the process started.
- The sixth column is the `TTY`, the controlling terminal for the process. Note that system processes (daemons) display a question mark (?).
- The seventh column is the `TIME`, the cumulative execution time for the process.
- The eighth column is the `CMD`, the command name, options, and arguments.

## List Process Trees in Oracle Linux

- The process tree (`pstree`) command lists the running processes, rooted at either `systemd` or PID.

```
$ pstree [-options] [PID | user]
```

```
$ ps -f
UID PID PPID C STIME TTY TIME CMD
oracle 2274 2262 0 14:47 pts/0 00:00:00 /bin/bash
oracle 2308 2274 0 14:47 pts/0 00:00:00 ps -f
$ pstree 2262
gnome-terminal---bash---pstree
 |---gnome-pty-help
 |---3*[{gnome-terminal-}]
```

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



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Listing Process Trees in Oracle Solaris

- The process tree (`ptree`) command lists the running processes, rooted at either `svc.startd` or PID.

```
$ ptree [PID | user]
```

```
$ ps -f
 UID PID PPID C STIME TTY TIME CMD
oracle 1770 1769 0 14:47 pts/1 0:00 ps -f
oracle 1769 1766 0 14:47 pts/1 0:00 /usr/bin/bash
$ ptree 1766
1766 /usr/bin/gnome-terminal -x /bin/sh -c cd '/home/oracle' && exec $S
 1769 /usr/bin/bash
 1771 ptree 1769
```

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



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Quiz



Which of the following is not a process attribute?

- a. UID
- b. GID
- c. PS
- d. PID



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Terminating a Process

- There might be times when you need to terminate an unwanted process.
- A process might have entered an endless loop, or it might be hung.
- You can kill or stop any process that you own.
- You can use the following two commands to terminate one or more processes:
  - `kill PID [PID ...]`
  - `pkill -l [ processname | regex_pattern ]`
- The `kill` and `pkill` commands send signals to processes directing them to terminate.
- Each signal has a number/value, name, and an associated event.
- For more information about signal values:
  - In Oracle Linux, use `man 7 signal`.
  - In Oracle Solaris, use `man -s3c signal`.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

A portion of the output from the `man 7 signal` command used in Oracle Linux.

| Signal  | Value    | Action | Comment                                                                 |
|---------|----------|--------|-------------------------------------------------------------------------|
| SIGHUP  | 1        | Term   | Hangup detected on controlling terminal or death of controlling process |
| SIGINT  | 2        | Term   | Interrupt from keyboard                                                 |
| SIGQUIT | 3        | Core   | Quit from keyboard                                                      |
| SIGILL  | 4        | Core   | Illegal Instruction                                                     |
| SIGABRT | 6        | Core   | Abort signal from <code>abort(3)</code>                                 |
| SIGFPE  | 8        | Core   | Floating point exception                                                |
| SIGKILL | 9        | Term   | Kill signal                                                             |
| SIGSEGV | 11       | Core   | Invalid memory reference                                                |
| SIGPIPE | 13       | Term   | Broken pipe: write to pipe with no readers                              |
| SIGALRM | 14       | Term   | Timer signal from <code>alarm(2)</code>                                 |
| SIGTERM | 15       | Term   | Termination signal                                                      |
| SIGUSR1 | 30,10,16 | Term   | User-defined signal 1                                                   |
| SIGUSR2 | 31,12,17 | Term   | User-defined signal 2                                                   |
| SIGCHLD | 20,17,18 | Ign    | Child stopped or terminated                                             |
| SIGCONT | 19,18,25 | Cont   | Continue if stopped                                                     |
| SIGSTOP | 17,19,23 | Stop   | Stop process                                                            |
| SIGTSTP | 18,20,24 | Stop   | Stop typed at terminal                                                  |
| SIGTTIN | 21,21,26 | Stop   | Terminal input for background process                                   |
| SIGTTOU | 22,22,27 | Stop   | Terminal output for background process                                  |

The signals `SIGKILL` and `SIGSTOP` cannot be caught, blocked, or ignored.

## Terminating a Process: kill Command

- You can terminate any process by sending the appropriate signal to the process.
- The `kill` command sends a Termination signal (SIGTERM signal 15) by default to one or more processes.

```
$ kill [-signal] PID [PID ...]
```

**Note:** The `kill` command terminates only those processes that you own.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

The `kill` command sends signal 15, the termination signal, by default. This signal causes the process to terminate in an orderly manner.

You need to know the PID of the process before you can terminate it. You can use either the `ps` or `pgrep` command to locate the PID of the process. Also, you can terminate several processes at the same time by entering multiple PIDs on a single command line.

**Note:** The `root` user/role can use the `kill` command on any process.

## Common Signals and Their Uses

- HUP (Hangup - SIGHUP signal 1) is sent to all of the subordinate child processes when the parent process is terminated.
- To terminate a command-line command you usually use a Ctrl + C keyboard command, which sends an Interrupt (SIGINT signal 2), causing it to be interrupted.
- In place of the `exit` command used to exit a terminal session, you can use a Ctrl + D keyboard command, which sends a Quit (SIGQUIT signal 3) to quit the terminal session.
- To stop the command-line command execution, you can use a Ctrl + Z keyboard command to send a Stop (SIGTSTP signal 19) to stop/suspend the foreground execution.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Every `spawn` (`ed`) or `fork` (`ed`) child process (`PID`) has an owning parent process (`PPID`) as seen in the `ps -f` command output. If the parent process is terminated, all the subordinate child processes must also be terminated. In UNIX and Linux, this is called a Hangup, detected on a controlling terminal (SIGHUP signal 1), and is sent to all of the subordinate child processes when the parent process is terminated.

Depending on the environment, a `ping` command can continue forever. To terminate the `ping` command, you usually use a Ctrl + C keyboard command, which sends an Interrupt from the keyboard (SIGINT signal 2) to `ping` the command, causing it to be interrupted.

In place of the `exit` command used to exit a terminal session, you can use a Ctrl + D keyboard command, which sends a Quit from the keyboard (SIGQUIT signal 3) to quit the terminal session.

All command-line commands execute in the foreground unless they are submitted followed by an & (ampersand). To stop the command-line command execution, you can use a Ctrl + Z keyboard command to send a Stop typed at the terminal (SIGTSTP signal 19) to stop the foreground processing. Then use a `bg %n` command to resume the command in the background.

## Terminating a Process: kill Command

Use the `kill` command to terminate the `dtmail` process.

```
ps -e | grep mail
 215 ? 00:00:03 sendmail
12047 ? 00:00:10 dtmail
kill 12047
ps -e | grep mail
 215 ? 00:00:03 sendmail
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

However, there are processes that should not be terminated, such as the `svc.startd`, `init`, or `systemd` processes. Killing such processes can result in a system crash.

**Note:** The `root` (superuser) can kill any process in the system.

## Terminating a Process: pgrep and pkill Commands

- One can use the `pgrep processname` command to identify the ProcessName or a `regex_pattern` to be killed, and then use the `pkill` command to kill them.

```
$ sleep 500 &
[1] 4378
$ pgrep sleep
4378
$ pgrep -l sleep
4378 sleep
$ pkill sleep
[1]+ Terminated sleep 500
```

- The `pkill` command requires you to specify the ProcessName or a `regex_pattern` instead of the PID of the process.
- For more information about the `pgrep` and `pkill` command options, see the `pgrep` man page.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Terminating a Process: pkill Command

Use the `pkill` command to terminate the `dtmail` process.

```
pkill dtmail
pgrep -l mail
215 ? 00:00:03 sendmail
```



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Forcefully Terminating a Process: Signal 9 (SIGKILL)

- Some processes ignore the default Termination signal (SIGTERM signal 15) that the `kill` command sends.
- If a process does not respond to the Termination signal 15, you can force it to terminate by sending the Kill signal (SIGKILL signal 9) with either the `kill` or `pkill` commands.

```
$ kill -9 PID
or
$ pkill -9 [processname | regex_pattern]
```

**Note:** Sending the Termination signal 15 does not necessarily kill a process gracefully. Only if the signal is caught by the process, it cleans itself up in an orderly fashion and dies. If not, it just dies.



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

**Caution:** Use the `kill -9` command only when necessary. When you use the `kill -9` command on an active process, the process terminates instantly. Sending signal 9 on processes that control databases or programs that update files could cause data corruption.

## Quiz



Ordinary users can only kill processes they own.

- a. True
- b. False



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Summary

In this lesson, you should have learned how to:

- Describe a process and its attributes
- Manage system process



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

## Practice 7: Overview

This practice covers the following topics:

- 7-1: Controlling System Processes



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 7 in your Activity Guide.

# Using Advanced Shell Features in Shell Scripts

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Objectives

After completing this lesson, you should be able to:

- Use advanced shell features
- Write shell scripts



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Agenda

- Using advanced shell features
- Writing shell scripts



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Jobs in the bash Shell

- A job is a process, which the shell manages.
- Each job is assigned a sequential job ID. A job is a process, therefore, each job has an associated process ID (PID).
- There are three types of job statuses:
  - Foreground
  - Background
  - Stopped

**Note:** Other shells support job control, except the Bourne shell.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

There are three types of job statuses:

- **Foreground:** When you enter a command in a terminal window, the command occupies that terminal window until it completes. This is a foreground job.
- **Background:** When you enter an ampersand (&) symbol at the end of a command line, the command runs without occupying the terminal window. The shell prompt is displayed immediately after you press Return. This is a background job.
- **Stopped:** If you press `Ctrl+Z` while a foreground job is running, or enter the `stop` command for a background job, the job stops. This job is called a stopped job.

## Job Control Commands

- Job control commands enable you to place jobs in the foreground or background, and to start or stop jobs.
- The following table describes the job control commands:

| Option                              | Description                                                                                                        |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| <code>Ctrl+Z</code><br>(SIGTSTP 19) | Stops the foreground job and places it in the background as a stopped job                                          |
| <code>jobs</code>                   | Lists all jobs and their job IDs                                                                                   |
| <code>bg [%n]</code>                | Places the current stopped job or the specified job ID in the <b>background</b> , where <i>n</i> is the job ID     |
| <code>fg [%n]</code>                | Brings the current or specified job ID from the background to the <b>foreground</b> , where <i>n</i> is the job ID |
| <code>kill %n</code>                | Deletes the job from the background, where <i>n</i> is the job ID                                                  |
| <code>kill -19 %n</code>            | Or, if signal 19 (SIGSTOP) is used, places the process associated with the job ID ( <i>n</i> ) in a stopped state  |

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The job control commands enable you to run and manage multiple jobs in a shell. However, you can use the job control commands only in the shell where the job was started.

There are two signal 19s, Stop process (SIGSTOP signal 19) that is usually associated with "kill -19 PID" and Stop typed at terminal (SIGTSTP signal 19), which is `Ctrl+Z` sent from the keyboard, both of which stop the process without killing it. Then using the two bash shell built-ins, `bg` (background) and `fg` (foreground), they send a signal Continue if stopped (SIGCONT signal 18) to continue processing either in the background or in the foreground.

## Running a Job in the Background

- To run a job in the background, you need to enter the command that you want to run, followed by an ampersand (&), which is a shell metacharacter, at the end of the command line.
  - For example, to run the sleep command in the background:

```
$ sleep 500 &
[1] 3028
```

- The shell returns the job ID, in brackets (which it assigns to the command), and the associated PID.

**Note:** The sleep command suspends the execution of a program for *n* seconds.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** With the job ID, you can use the job control commands to manage the job, whereas the kernel uses PIDs to manage jobs.

When a background job is complete and you press Return, the shell displays a message indicating the job is done.

```
[1]+ Done sleep 500
```

## Bringing a Background Job to the Foreground

- You can use the `jobs` command to list the jobs that are currently running or stopped in the background.

```
$ jobs
[1] + Running sleep 500 &
```

- You can use the `fg` command to bring a background job to the foreground.

```
$ fg %1
sleep 500
```

**Note:** The foreground job occupies the shell until the job is completed or stopped and placed into the background.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You can use the `Ctrl+Z` keys and `bg` command to return a job to the background. The `Ctrl+Z` keys stop the job, and place it in the background as a stopped job. The `bg` command runs the job in the background.

```
$ sleep 500
^Z
[1]+ Stopped sleep 500
$ sleep 500 &
[2] 4578
$ jobs
[1]+ Stopped sleep 500
[2]- Running sleep 500 &
$ bg %1
[1]+ sleep 500 &
$ jobs
[1]- Running sleep 500 &
[2]+ Running sleep 500 &
$ fg %1
sleep 500
-
```

**Note:** When you place a stopped job either in the foreground or background, the job resumes.

## Quiz



To run a job in the background, you need to enter the command that you want to run, followed by a pipe (|) symbol at the end of the command line.

- a. True
- b. False



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## The alias Command

- An `alias` is a shorthand shell notation that allows you to customize and abbreviate commands.

```
$ alias aliasname="command_string"
```

- If the first word on the command line is an `aliasname`, the shell replaces that word with the text of the alias.
- The shell maintains a list of aliases that it searches when a command is entered.
- The following rules apply while creating an alias:
  - There can be no whitespace on either side of the equal sign.
  - The command string must be quoted if it includes any options, metacharacters, or whitespace.
  - Each command in a single alias must be separated by a semicolon.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The `alias` command is shell notation built into the bash shell. Aliases are also available in other shells.

## Command Sequence

- You can group several commands under a single *aliasname*.
- Individual commands are separated by semicolons.

```
$ alias info='uname -a; id; date'
$ info
SunOS s11-server1 5.11 11.3 i86pc i386 i86pc
uid=60016(oracle) gid=100(oracle)
Fri Feb 10 15:22:47 UTC 2017
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

This alias is created using a pipe (|) to redirect the output of the ls -l command to the less command. When the new alias is invoked, a directory list appears.

```
$ alias ll='ls -l | less'
$ cd /usr
$ ll
total 136
drwxrwxr-x 2 root bin 1024 Feb 13 18:33 4lib
drwx----- 8 root bin 512 Feb 13 18:14 asset
drwxrwxr-x 2 root bin 7168 Feb 13 18:23 bin
drwxr-xr-x 4 bin bin 512 Feb 13 18:13 ccs
drwxrwxr-x 5 root bin 512 Feb 13 18:28 demo
```

## Predefined Aliases

- In Oracle Linux, the GNU bash shell contains several predefined system aliases.
- You can display these predefined aliases by using the `alias` command.

```
[oracle@ol7-server1 ~]$ alias
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias lss='ls --color=auto'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
```

**Note:** The `alias` command displays both system- and user-defined aliases.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle Linux provides several system-defined aliases, while Oracle Solaris does not contain any system-defined aliases.

## User-Defined Aliases

- User-defined aliases are defined by a user, usually to abbreviate or customize frequently used commands.
- The `history` command is aliased as `h` by using the `alias` command in the following code:

```
$ alias h=history
$ h
278 cat /etc/passwd
279 pwd
280 cp /etc/passwd /tmp
281 ls ~
282 alias h=history
283 h
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Using the `rm`, `cp`, and `mv` commands can inadvertently result in loss of data. As a precaution, you can alias these commands with the interactive option. The `rm` command is aliased with the `-i` option as coded as follows:

```
$ alias rm='rm -i'
$ rm dat1
rm: remove dat1: (yes/no) ? N
```

Similarly, creating a `cp -i` and `mv -i` alias ensures that the shell prompts you for confirmation before overwriting existing files.

## Deactivating an Alias

- You can temporarily deactivate an alias by placing a backslash (\), the shell metacharacter **escape**, in front of the alias on the command line.
- In the following code, the backslash prevents the shell from looking in the alias list. This allows the shell to run the original `rm` command to remove the `file1` file.

```
$ rm file1
rm: remove file1 (yes/no)? n
$ \rm file1
$ ls file1
file1: No such file or directory
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Removing an Alias

- The `unalias` command removes aliases from the alias list.

```
$ unalias aliasname
```

- The `h` alias that was created earlier is removed by using the `unalias` command.

```
$ unalias h
$ h
bash: h: not found
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** To pass new aliases to every shell invoked, place the alias definition in your `~/.bashrc` file.

## Quiz



Which of the following rules does not apply while creating an alias?

- a. There can be no space on either side of the equal sign.
- b. The backslash (\) is always placed in front of the alias.
- c. The command string in an alias must be quoted if it includes any options, metacharacters, or whitespace.
- d. Each command in a single alias must be separated with a semicolon.



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Shell Functions

- Functions, which is a powerful feature of shell programming, is a group of commands organized by common functionality.
- There can be hundreds of predefined shell functions.
- These easy-to-manage units, a function when called returns a single value with no additional output.
- Using a function involves two steps:
  - Defining the function
  - Invoking the function



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** Shell functions and aliases are different on two counts. First, aliases do not take arguments as functions do. Second, if a command name is defined as a function and an alias, the alias takes precedence.

To display a list of all functions, use the following command:

```
$ typeset -f
... (output truncated)
function list ()
{
ls -al | wc -l;
}
function num ()
{
who | wc -l;
}
```

To display just the function names, use the following command:

```
$ typeset -F
... (output truncated)
declare -f list
declare -f num
```

## Defining a Function

- A function is defined by using the following general syntax:

```
function functionname [()] { compound-command [redirections]; }
```

- To define a function called `num` that displays the total number of users currently logged in to the system:

```
$ function num { who | wc -l; }
```

- The `num` function runs the `who` command, whose output is redirected to the `wc` command.

- To remove a function, use the `unset -f` command:

```
$ unset -f num
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Functions are not only useful in shell scripts but they can be useful when used in command-line situations where an alias is unusable. For demonstration, the shell functions are run on the command line to illustrate how the functions perform.

To define or create a function called `list` that displays the total number of subdirectories and files in the current directory:

```
$ function list { ls -al | wc -l; }
$ list
34
```

The `list` function executes the `ls` command, whose output is redirected (piped) to the `wc` command.

**Note:** In UNIX and Linux, *whitespace* is a delimiter. Therefore, a space must appear after the opening brace and before the closing brace.

## Invoking a Function

You can invoke a function by merely entering the function name on the command line or within the shell script.

```
$ num
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Shell Options

- Options are switches that control the behavior of the shell. In the `bash` shell, there are about 27 different options.
- Options are of the `Boolean` data type, which means that they can be either `on` or `off`.
- To show the current option settings, enter:

```
$ set -o
```

- To turn on an option, enter:

```
$ set -o option-name
```

- To turn off an option, enter:

```
$ set +o option-name
```

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** The `set -o` and `set +o` options can change only a single option setting at a time. There are over a couple of dozen options.

## Activating the noclobber Shell Option

- Redirecting standard output (`stdout`) to an existing file overwrites the previous file's content that results in data loss.
- This process of overwriting existing data is known as *clobbering*.
- To prevent an overwrite from occurring, the shell supports a `noclobber` option.
- When the `noclobber` option is set, the shell refuses to redirect standard output to the existing file and displays an error message on the screen.
- The `noclobber` option is activated in the shell by using the `set -o` command.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The `noclobber` option is activated in the shell by using the `set` command.

```
$ set -o noclobber
$ set -o | grep noclobber
noclobber on
$ ps -ef > file_new
$ cat /etc/passwd > file_new
bash: file_new: cannot overwrite existing file
```

**Note:** To ensure that the `noclobber` option is available to every shell invoked, place the `set -o noclobber` command in your `~/.bashrc` file.

## Deactivating the noclobber Shell Option

- To deactivate the `noclobber` option, enter the following commands:

```
$ set +o noclobber
$ set -o | grep noclobber
noclobber off
```

- To temporarily deactivate the `noclobber` option, use the `>|` deactivation syntax on the command line:

```
$ ls -l >| file_new
```

**Note:** There is no space between the `>` and `|` on the command line. The `noclobber` option is ignored for this command line only, and the contents of the file are overwritten.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Quiz



Which of the following syntaxes can be used to turn off an option?

- a. \$ set +o option-name
- b. \$ set -o e
- c. \$ set -o option-name



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Agenda

- Using advanced shell features
- Writing shell scripts



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

## Shell Scripts

- A shell script is a text file that contains a sequence of commands and comments for a UNIX-like OS, and is designed to be run by the UNIX **shell**, a command-line interpreter.
- Shell scripts are often used to automate repeating command sequences, such as services that start or stop on system startup or shutdown.
- There can be many shell scripting languages, for example, Perl, PHP, Tcl, and so on. However, for this lesson, we will focus on the default shell `bash`.
- Users with little or no programming experience can create and run shell scripts.
- You can run the shell script by simply entering the name of the shell script on the command line.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** A shell script can be executed on the command line if it is made executable with the "`chmod u+rwx script-name`" permission. Then if the script is not found in "`echo $PATH`", you will need to preface script with `./script-name`. The `./` says look in the current working directory. Nonexecutable scripts can be executed by using the appropriate-script-interpreter `script-name`, for example, `bash script-name`.

## Determining the Shell to Interpret and Execute a Shell Script

- Oracle Solaris and Oracle Linux support various shell scripting languages, such as GNU Bash, Bourne, Korn, C, Perl, PHP, and others.
- The first line of a shell script identifies the shell program that interprets and executes the commands in the script.
- The first line should always begin with the symbols `#!` (called a `shebang`) followed immediately by the absolute pathname of the shell program used to interpret the script.

```
#!/full-pathname-of-shell
```

- The first line for a bash shell script is as follows:

```
#!/usr/bin/bash
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Note:** Shell scripts are rarely compiled into binary form.

## Creating a Shell Script

- To create a shell script, you need a text editor.
- A text editor is a program that reads and writes text files.
- The following code is a simple shell script:

```
#!/usr/bin/bash
This is my first shell script.
echo "Hello World!"
```

- The first line of the script indicates the shell program that interprets the commands in the script. In this case, it is `/usr/bin/bash`.
- The second line is a comment. Everything that appears after a hash (#) symbol is ignored by `bash`.
- The last line is the `echo` command, which prints what is displayed.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Alternatively, executing a shell script from the command line using: `here document`.

The "Hello World" script can be directly keyed into the command-line interface.

```
1 $ #!/usr/bin/bash << EOF
2 > # This is my first shell script.
3 > echo "Hello World!"
4 > EOF
5 Hello World!
6 $
```

Observe at the end of the line 1 the `<<` delimiter; this is called "`here document`." The delimiter in this case is the string `EOF`. Then again on line 4 you see the same delimiter `EOF` ending/closing the "`here document`." These literals/delimiters are used to encapsulate the code of the script. You can use any delimiting string you want like "`ThisIsTheEnd`", although the delimiting string "`EOF`" is fairly common.

## Executing a Shell Script

- After the shell script is created, you can run it.
  - To run a shell script, the user must have execute permissions.
    - To grant read and execute permissions to the user so that you can execute the `mycmd` shell script, use the `chmod` command:
- ```
$ chmod u+rwx mycmd
```
- A shell script is executed by just calling out the script name on the command line.
 - To run the `mycmd` script in the current directory, enter `./mycmd` on the command line.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: When a shell script is running, any applied changes occur in the subshell or child process. A subshell cannot change the values of a variable in the parent shell, or its working directory.

```
$ cat myvars
echo running myvars
FMHOME=/usr/frame
MYBIN=/export/home/oracle/bin
echo $FMHOME
$ ls -l myvars
-rw-r--r-- 1 oracle oracle 65 Feb 15 16:14 myvars
$ chmod u+xt myvars
$ ls -l myvars
-rwxr--r-- 1 oracle oracle 65 Feb 15 16:14 myvars
$ ./myvars
running myvars
/usr/frame
$ grep $FMHOME
$
```

The `grep` command returns nothing because `FMHOME` is not defined in the parent shell.

Comments in a Shell Script

- A comment is a textual description of the script and the lines within the script file.
- Comments are always preceded by a hash (#) symbol.

```
# This is a comment inside a shell script  
ls -l # lists the files in a directory
```

- Whenever a shell encounters a hash (#) symbol in a script file, the line following it is ignored by the shell.
- The addition of comments in a shell script file does not affect the execution of the script unless a syntactical error is introduced when the comments are added.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Positional Parameters in a Shell Script

- You can pass command-line arguments to a shell script while it is running.
- As you pass these arguments on the command line, the shell stores the first parameter after the script name into variable \$1, the second parameter into variable \$2, and so on.
- These variables are called *positional* parameters.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following set of commands illustrates how arguments are passed to the greetings script.

1. View the greetings script.

```
$ cat greetings
#!/usr/bin/sh
echo $1 $2 #echo the first two parameters passed
```

2. Add execute permissions to greetings.

```
$ chmod u+x greetings
```

3. Run greetings while passing the three strings/parameters/arguments:
"hello world a-third-parameter".

```
$ greetings hello world a-third-parameter
hello world
```

Note: Only the first two parameters are displayed.

Quiz



To run a shell script, the user must have write permissions.

- a. True
- b. False



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Checking the Exit Status

- Exit status is a numeric value that indicates the success or failure of a command.
 - A value of zero indicates success.
 - A nonzero value indicates failure.
 - This nonzero value can be any integer in the range of 1–255.
- All commands in the UNIX and Linux environments return an exit status, which is held in the read-only shell variable `?`.
- A developer can use exit status values to indicate different error situations.
 - The exit status value can be set inside a shell script with the `exit=##` command.
- To check the value of exit status, use the `echo` command and the variable expansion dollar sign (`$`):

```
$ echo $?
0
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The exit status of the last command run in the foreground is held in the "?" special shell variable, and can be printed by using the `echo` command.

```
$ grep other /etc/group
other::1:
$ echo $?
0          # true
$ grep others /etc/group
$ echo $?
1          # false
$ ls -l somebadfilename
ls: cannot access somebadfilename: No such file or directory
$ echo $?
2          # false
```

The test Command

- The shell built-in `test` command is used for testing conditions.
- The `test` command can be written as a test expression or written using the `[expression]` special notation.
- The `test` command is also used for evaluating expressions, such as the following:
 - Variable values
 - File access permissions
 - File types
- There are several types or categories of bash shell test comparison operators:
 - Integer/Arithmetic test comparison operators
 - String test comparison operators
 - File test comparison operators
 - And there are other test comparison operators



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Using the `test` command, test whether the value of the `LOGNAME` variable is `oracle`.

```
$ echo $LOGNAME
oracle
$ test "$LOGNAME" = "oracle"
$ echo $?
0      # true
```

Test whether the value of the `LOGNAME` variable is `oracle` by using the `[expression]` notation.

```
$ echo $LOGNAME
oracle
$ [ $LOGNAME = someothername ]
$ echo $?
1      # false
```

Note: The double-quotes can be used to encapsulate whitespace, and provide isolation for the variables and values being tested. Plus, whitespace is needed within the brackets (`[]`) to separate the test operands.

Integer/Arithmetic test Comparison Operators

The Integer/Arithmetic test comparison operators use characters:

Operator	Meaning	Syntax
-eq	Equal to	["\$var1" -eq "\$var2"] Note1: ^ ^ ^ ^ required whitespace (IFS) Note2: While not required, the double-quotes (" ") provide excellent variable and value isolation and is considered a Best Practice.
-ne	Not equal to	["\$var1" -ne "\$var2"]
-le	Less than or equal to	["\$var1" -le "\$var2"]
-ge	Greater than or equal to	["\$var1" -ge "\$var2"]
-lt	Less than	["\$var1" -lt "\$var2"]
-gt	Greater than	["\$var1" -gt "\$var2"]



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Test the relationship between two integers `int1` and `int2` and literals by using the [expression] notation.

```
$ int1=1 ; int2=2
$ echo "int1 is equal to '$int1' and int2 is equal to '$int2'"
int1 is equal to '1' and int2 is equal to '2'
$ [ "$int1" -eq "$int2" ]
$ echo $?
1          # false
$ [ "$int1" -eq "1" ]
$ echo $?
0          # true
$ [ "$int2" -eq "2" ]
$ echo $?
0          # true
$ [ "$int1" -ne "$int2" ]
$ echo $?
0          # true
$ [ "$int1" -lt "$int2" ]
$ echo $?
0          # true
$ [ "$int1" -gt "$int2" ]
$ echo $?
1          # false
```

String test Comparison Operators

The String test comparison operators use symbols.

Operator	Meaning	Syntax
= or ==	Equal to	["\$str1" == "\$str2"]
		Note: While both = and == are usable as string comparison operators. The = is also used as an assignment operator. The == is considered a Best Practice.
!=	Not equal to	["\$str1" != "\$str2"]
-z	String is <i>null</i> , zero length	["\$str1" -z "\$str2"]
-n	String is not <i>null</i>	["\$str1" -n "\$str2"]
<	Sorts before	["\$str1" < "\$str2"]
>	Sorts after	["\$str1" > "\$str2"]



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Test the relationship between two strings strA and strB and literals by using the [expression] notation.

```
$ strA=abc ; strB=def
$ echo "strA is equal to '$strA' and strB is equal to '$strB'"
str1 is equal to 'abc' and str2 is equal to 'def'
$ [ "$strA" == "$strB" ]
$ echo $?
1          # false
$ [ "$strA" == "abc" ]
$ echo $?
0          # true
$ [ "$strB" == "def" ]
$ echo $?
0          # true
$ [ "$strA" != "$strB" ]
$ echo $?
0          # true
```

File test Comparison Operators

The File test comparison operators use characters. (Not a complete listing.)

Operator	Meaning	Syntax
-e (or -a) filename	File exists	[-e filename]
-f filename	File is a regular file	[-f filename]
-d filename	File is directory	[-d filename]
-c filename	File is a character device	[-c filename]
-b filename	File is a block device	[-b filename]
-h (or -L) filename	File is a symbolic link	[-h filename]
-r filename	File is a readable	[-r filename]
-w filename	File is a writeable	[-w filename]
-x filename	File is a executable	[-x filename]
-s filename	File size is bigger than zero bytes (not empty)	[-s filename]



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Test whether the filename lab/dante exists by using the [expression] notation.

```
$ pwd
/home/oracle
$ [ -e lab/dante ]
$ echo $?
0          # true
```

Test whether the filename lab/dante2 exists by using the [expression] notation.

```
$ pwd
/home/oracle
$ [ -e lab/dante2 ]
$ echo $?
1          # false
```

Note: If the value of the exit status is 0 (zero – true), the filename exists; if the value is 1 (false), the filename does not exist.

Test whether the filename lab/dante is a regular file by using the [expression] notation.

```
$ ls -l lab/dante
-rw-r--r--. 1 oracle oracle 1319 ... lab/dante
$ [ -f lab/dante ]
$ echo $?
0          # true
```

Using the test Command in an if Statement

- The `test condition` command or `[expression]` special notation often follows the `if` statement.
- The `test` command evaluates a condition and, if the result of the test is true, it returns an exit status of zero.
- If the result of the test is false, the `test` command returns a nonzero exit status.

```
if test condition
then
  command
...
fi
or
if [ expression ]
then
  command
...
fi
```

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Test whether the value of the `LOGNAME` variable is `oracle` by using the `[expression]` notation.

```
$ echo $LOGNAME
oracle
$ if [ "$LOGNAME" == "oracle" ]
> then      # true path
> echo $?
> echo "Hello $LOGNAME"
> fi
0
Hello oracle
```

How to Test/Debug a Shell Script

There are several bash shell command-line options, which can also be included in your shell scripts on the line with the shebang after the script interpreter.

- The `-x` option described as a *debugger*
- The `-v` option described as *verbose*
- The `-n` option described as a *syntax checker*

```
$ bash -xv scriptname  
or  
$ cat scriptname  
#!/usr/bin/bash -xv  
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Conditional Expressions

The shell provides the following special expressions that enable you to run a command based on the success or failure of the preceding command.

- The `&&` operator
- The `||` operator
- The `if` statement
- The `case` statement



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The && Operator

The `&&` (and) operator ensures that the second command is run only if the preceding command succeeds, both commands succeed.

```
$ mkdir $HOME/newdir && cd $HOME/newdir
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Test the `&&` (and) operator with a second command.

```
$ pwd  
/home/oracle  
$ mkdir somenewdir && cd somenewdir  
$ pwd  
/home/oracle/somenewdir
```

Running the same series of commands again, produces entirely different results.

```
$ pwd  
/home/oracle  
$ mkdir somenewdir && cd somenewdir  
mkdir: cannot create directory 'somenewdir': File exists  
$ pwd  
/home/oracle
```

Note: Because the `mkdir` command failed, the `cd` command is not attempted.

The || Operator

The || (or) operator ensures that the second command is run only if the preceding command fails.

```
$ mkdir /usr/tmp/newdir || mkdir $HOME/newdir
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Test the || (or) operator with a second command.

```
$ pwd  
/home/oracle  
$ mkdir somenewdir || cd somenewdir  
mkdir: cannot create directory 'somenewdir': File exists  
$ pwd  
/home/oracle/somenewdir
```

Note: Even though the `mkdir` command failed, the `cd` command completes successfully.

The if Statement

- The `if` statement evaluates the exit status of a command and initiates additional actions based on the return value.

```
$ if [ command | test ];
> then                      # on true execute
>   command1
>   ...
> else                        # on false execute
>   commandn
>   ...
> fi
```

- If the exit status is zero (true), any commands that follow the `then` statement are run.
- If the exit status is nonzero (false), any commands that follow the `else` statement are run.

Note: The `if` statement is often used with the `test` command.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Display a greetings message by using an `if` statement.

```
$ id
uid=101(frame) gid=1(other)
$ if test $LOGNAME = root
>   then                      # true path
>   echo Hello System Administrator
> else                        # false path
>   echo $?
>   echo "Hello " $LOGNAME
> fi
1
Hello frame
```

The `if` Statement Additional Syntax

Within the `if` statement, there can be multiple nested `if` test using the `elif` syntax in place of an `else`.

```
$ if [ command1 | test1 ];
> then                                # on true execute
>   commandn
>   ...
> elif [ command2 | test2 ];  # on false run a second nested test
>   then                                # on true from the second nested test execute
>     commandn
>     ...
>   else                                # on false from the second nested test execute
>     commandn
>     ...
> fi
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Display the greetings message by using an `if` statement (which failed) with an `elif` statement and a second test.

```
$ whoami
oracle
$ if [ $LOGNAME == root ]
>   then      # true path
>   echo Hello System Administrator
> elif [ $LOGNAME == oracle ] # false path and a second test
>   then      # true path second test
>   echo $?
>   echo "Hello " $LOGNAME
> else      # false path from the second test
>   echo "Don't know LOGNAME"
> fi
0
Hello oracle
```

The case Statement

The `case` command compares a single value against other values, and runs a command or group of commands when a match is found.

```
$ case value in
> pat1)
> command1
> ...
> ;;
> patn)
> commandn
> ...
> ;;
> *)      # The * is a catch-all
> echo "Usage: $0 { pat1 | patn }"
> exit 3
> ;;
> esac
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

After a match is found for `start` or `stop` and the respective commands are run, no other patterns are checked.

```
#!/sbin/sh
# Copyright 2009 Oracle Corporation All rights reserved.
# Use is subject to license terms.#
# ident"@(#)volmgmt1.703/12/09 SMI"
case "$1" in
'start')
    if [ -f /etc/vold.conf -a -f /usr/sbin/vold -a \
    "${_INIT_ZONENAME:='/sbin/zonename'}" = "global" ]; then
        echo 'volume management starting.'
        /usr/sbin/vold >/dev/msglog 2>&1 &
    fi
    ;;
'stop')
    /usr/bin/pkill -x -u 0 vold
    ;;
*)
    echo "Usage: $0 { start | stop }"
    exit 1
    ;;
esac
```

Looping Constructs

Many programming/scripting languages provide structures to iterate through a list of objects. The bash shell provides the following three loop structures.

- The `for` loop statement
- The `while (true)` loop statement
- The `until (true)` loop statement



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The `for` Loop Statement

- The `for` command enables you to repeat a command or group of commands in a loop.

```
$ for arg in [ command | test ]
> do
>   commandn
>   ...
> done
```

- The `for` command evaluates the exit status of the `in` operation that follows it.
 - If the exit status is zero, any instructions that follow the `do` statement are run, `command` or `test` is rerun, and the exit status rechecked.
 - If the exit status is nonzero, the loop terminates.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

A `for` loop allows one to repeat the section of code encapsulated by `do done`. This simple `for` counts up from 1 to 5, and then exits.

```
$ for (( i=1 ; i <= 5 ; i++ ))
>   do
>     echo $i
>   done
1
2
3
4
5
```

To reverse the order, you can count down from 5 to 1 then exit.

```
$ for (( i=5 ; i >= 1 ; i-- ))
>   do
>     echo $i
>   done
5
4
3
2
1
```

Shifting Positional Parameters in a Loop

- While passing command-line arguments, the Bourne (`sh`) shell accepts only a single number after the `$` sign (`$0-$9`).
- An attempt to access the value in the tenth argument using the notation `$10` results in the value of `$1` followed by a zero (0).
- Both the Korn (`ksh`) shell and Bash (`bash`) shell can access the 10th parameter directly with the value of the 10th argument `${10}`. However, that could become very cumbersome in a loop.
- The `shift` command enables you to shift your positional parameter values back by one position when processing the positional parameters in a loop.
 - The value of the `$2` parameter becomes assigned to the `$1` parameter.
 - Therefore, there is **no limit** on the number of positional parameters that can be passed to a shell script.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Use the `set` command to assign values for six positional parameters as follows. Then inside the `do done` construct, use the `shift` command to shift through those assigned values.

```
$ set 5 4 3 2 1 Blastoff
$ echo $@
5 4 3 2 1 Blastoff
$ for arg in $@
> do
>   (( i = i + 1 ))
>   echo "Positional parameter $1 the number of times through the loop $i"
>   shift
> done
Positional parameter 5 the number of times through the loop 1
Positional parameter 4 the number of times through the loop 2
Positional parameter 3 the number of times through the loop 3
Positional parameter 2 the number of times through the loop 4
Positional parameter 1 the number of times through the loop 5
Positional parameter Blastoff the number of times through the loop 6
```

The `while` (True) Loop Statement

- The `while` command enables you to repeat a command or group of commands in a loop.

```
$ while [ command | test ]  
> do  
>   commandn  
>   ...  
> done
```

- The `while` command evaluates the exit status of the `command` or `test` command that follows it.
 - If the exit status is zero, any instructions that follow the `do` statement are run, `command` or `test` is rerun, and the exit status rechecked.
 - If the exit status is nonzero, the loop terminates.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Use the `set` command to assign values to the positional parameters as follows:

```
$ set this is a while loop  
$ echo $@  
this is a while loop  
$ while [ $# -gt 0 ]  
> do  
>   echo $1  
>   shift  
> done  
this  
is  
a  
while  
loop
```

The until (True) Loop Statement

- The `until` command enables you to repeat a command or a group of commands in a loop:

```
$ until [ command | test ]
> do
>   commandn
>   ...
> done
```

- The `until` command evaluates the exit status of the `command` or the `test` command that follows it.
 - If the exit status is nonzero, any instructions that follow the `do` statement are run, `command` or `test` is rerun, and the exit status rechecked.
 - If the exit status is zero, the loop terminates.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Use the `set` command to assign values to the positional parameters as follows:

```
$ set this is an until loop
$ echo $@
this is a until loop
$ until [ $# -le 0 ]
> do
>   echo $1
>   shift
> done
this
is
an
until
loop
```

Quiz



Which of the following evaluate the exit status of a command and initiate additional actions based on the return values?

- a. The `case` statement
- b. The `test` command
- c. The `if` statement
- d. The `while` statement



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Summary

In this lesson, you should have learned how to:

- Use advanced shell features
- Write shell scripts



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Practice 8: Overview

This practice covers the following topics:

- 8-1: Using advanced Bash shell functionality
- 8-2: Using shell scripts
- 8-3: Using the `test` command and conditional expressions



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find the tasks for Practice 8 in your Activity Guide.

Unauthorized reproduction or distribution prohibited. Copyright 2017, Oracle and/or its affiliates.

Oracle University and Canadian Business School use only.

Archiving, Compressing, and Performing Remote File Transfers

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red rectangular background.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Archive and retrieve files
- Compress, view, and uncompress files
- Perform remote connections and file transfers



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Agenda

- Archiving and retrieving files
- Compressing, viewing, and uncompressing files
- Performing remote connections and file transfers



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

File Archival: Introduction

- To safeguard your files and directories, you can create a copy of all the files and directories in your file system.
- This copy is a repository of files and directories and is called an archive.
- The archive serves as backup in the event of data loss.
- You can create an archive on a storage device, such as a remote disk or tape or you can send your archive to the cloud.
- Of the many commands, the `tar` command is the most commonly used for creating and retrieving archived files.

Note: It is a good practice to use relative path names to archive files.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: `cpio` is yet another archival program. Unlike `tar`, which automatically recurses subdirectories, `cpio` reads a list of files and directories from `stdin`, creates the archive, and writes the archive to `stdout`.

The `tar` Command

- The `tar` command creates, adds, deletes, lists, or extracts files in a tape archive file.

```
$ tar [options] archivefile filenames
```

- The output of using a `tar` command is a `tar` file.
- The default output location for a `tar` file in UNIX and Linux is `stdout`.
- For more information about the `tar` command options, see the `tar` man page.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: The `tar` command in Oracle Solaris and Oracle Linux strips the leading "/" symbol automatically. This means that the files are now extracted to the current directory and not into the root directory.

The Common `tar` Command Options

Option	Description
<code>c</code>	Creates a new <code>tar</code> file
<code>t</code>	Lists the table of contents of the <code>tar</code> file
<code>x</code>	Extracts files from the <code>tar</code> file
<code>f</code>	Specifies the archive file or tape device.
<code>v</code>	Executes in verbose mode; writes to the standard output
<code>h</code>	Follows symbolic links as standard files or directories
<code>z</code>	Compresses and extracts files and directories by using <code>gzip</code>
<code>j</code>	Compresses and extracts files and directories by using <code>bzip2</code>



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The table in the slide describes some of the commonly used `tar` command options. For a detailed explanation of the `tar` command and its options, read the `tar` man page or `tar --help`.

Creating a `tar` Archive

- You can use the `tar` command to create an archive file containing multiple files or directories on a disk or in a single file.
- The following example shows you how to archive your home directory on a disk:

```
$ tar [-]cvf /dev/rmt/0 .
a ./ 0 tape blocks
a ./rhosts 1 tape blocks
... (output truncated)
```

- The following example shows you how to archive multiple files into an archive file called `files.tar`:

```
$ tar [-]cvf files.tar file1 file2 file3
a file1 2K
a file2 1K
a file3 1K
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The “a” at the beginning of the output shows that a file has been **added** to the archive.

Note: The `tar` command was one of the earliest UNIX utilities created, and the use of the “-” (a hyphen or dash) with the command’s options had not yet been finalized. The `tar` command is one of a very few commands that does not require the use of “-”, and therefore, the syntax shows an optional `[-]`.

Viewing the Table of Contents of a tar Archive

- You can view the names of all the files that have been written directly to a disk or an archive file.
- To view the table of contents of Oracle's home directory on the disk, enter the following command:

```
$ tar [-]tf /dev/rmt/0
/.rhosts
./dante
./fruit
... (output truncated)
```

- To view the verbose content of the files.tar archive file, enter the following command:

```
$ tar [-]tvf files.tar
-rw-rw-r-- oracle/oracle 1610 ... file1
-rw-rw-r-- oracle/oracle 105 ... file2
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Extracting a tar Archive

- You can retrieve or extract the entire contents of an archive or a single file that was written directly to a disk device or to an archive file.
- To retrieve all the files from the disk archive, enter the following command:

```
$ tar [-]xvf /dev/rmt/0
x ., 0 bytes, 0 tape blocks
x ./rhosts, 2 bytes, 1 tape blocks
... (output truncated)
```

- To extract or restore a single file from the `files.tar` archive file, enter the following command:

```
$ tar [-]xvf files.tar file1
x file1, 1610 bytes, 4 tape blocks
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The “x” at the beginning of the output shows that a copy of the file was **extracted** from the archive.

Note: This output is from Oracle Solaris and the output from Oracle Linux may be slightly different.

Quiz



Which command do you use to view the table of contents of the archive file named file8.tar?

- a. tar xvf file8.tar
- b. tar cvf file8.tar
- c. tar tf file8.tar



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Agenda

- Archiving and retrieving files
- Compressing, viewing, and uncompressing files
- Performing remote connections and file transfers



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

File Compression

- With the enormous amount of enterprise data that is created and stored, there is a pressing need to conserve disk space and optimize data transfer time.
- There are various tools, utilities, and commands that are used for file compression. Some of the more commonly used commands are:
 - The `gzip` command
 - The `zip` command
 - The `bzip2` command



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Compressing a File: gzip Command

- Use the `gzip` command to compress files:

```
$ gzip [options] filename(s)
```

- For example, to compress a set of files, `file1`, `file2`, `file3`, and `file4`, enter the following command:

```
$ gzip file1 file2 file3 file4
$ ls *.gz
file1.gz file2.gz file3.gz file4.gz
```

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: The compressed files have a `.gz` extension.

Uncompressing a File: gunzip Command

- The `gunzip` or `gzip -d` command uncompresses a file that has been compressed by using the `gzip` command:

```
$ gunzip [options] filename
```

- To uncompress or decompress the `file1.gz` file, use the following command:

```
$ gunzip file1.gz  
or  
$ gzip -d file1.gz
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Viewing a Compressed File: zcat Command

- In Oracle Linux, the `zcat` command prints the uncompressed form of a compressed file to `stdout`.

```
$ zcat [options] filename
```

- To view the content of the `dante.gz` compressed file, enter the following command:

```
[oracle@ol7-server1 lab]$ zcat dante.gz | less
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
... (output truncated)
```

Note: The `zcat` command interprets the compressed data and displays the content of the file as if it were not compressed.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: The `zcat` command does not change the content of the compressed file. The compressed file remains on the disk in compressed form.

Viewing a Compressed File: `gzcat` Command

- In Oracle Solaris, the `gzcat` command displays the content of files that were compressed with the `gzip` command to stdout.

```
$ gzcat [options] filename
```

- To view the `file1.gz` file, use the following command:

```
[oracle@s11-server1:~/lab]$ gzcat file1.gz
The Achievers
Unconsciously or not, they divide their work totally
differently than the sustainers do. Certainly Achievers work
longer hours. New York magazine has published several surveys
on work needs which reveal that well-known typically work from
... (output truncated)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: The `gzcat` command does not change the content of the compressed file. The compressed file remains on the disk in compressed form.

Archiving and Compressing Multiple Files: `zip` Command

- The `zip` command archives and compresses multiple files into a single archive file, and is compatible with files created with `pkzip`.

```
$ zip [options] archivefile filename(s)
```

- To compress `file2` and `file3` into the `file.zip` archive file, enter the following:

```
$ zip file.zip file2 file3
adding: file2 (deflated 16%
adding: file3 (deflated 26%)
$ ls
file.zip
file2
file3
```

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

By default, the `zip` command adds the `.zip` extension to the compressed archive file if you do not assign a new file name with an extension.

Note: You can run the `zip` or `unzip` command on the command line to view a list of options used with each command.

Viewing and Uncompressing Archive Files: `unzip` Command

- The `unzip` command is used for listing the files and also for extracting the content of a compressed `.zip` file.

```
$ unzip [options] archivefile
```

- To uncompress the `file.zip` archive file, use the following command:

```
$ unzip file.zip
```

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Compressing a File: bzip2 Command

- Use the `bzip2` command to compress files.

```
$ bzip2 [options] filename(s)
```

- For example, to compress a set of files, `file1`, `file2`, `file3`, and `file4`, enter the following command:

```
$ bzip2 file1 file2 file3 file4
$ ls *.bz2
file1.bz2 file2.bz2 file3.bz2 file4.bz2
```

- For more information about the `bzip2` and `bunzip2` command options, see the `bzip2` man page.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: The compressed files have a `.bz2` extension.

Uncompressing a File: bunzip2 Command

- The `bunzip2` command uncompresses a file that has been compressed with the `bzip2` command.

```
$ bunzip2 [options] filename
```

- To uncompress the `file1.bz2` file, use the following command:

```
$ bunzip2 file1.bz2
```

- For more information about the `bunzip2` command options, see the `bunzip2` man page.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Viewing a Compressed File: bzcat Command

- The `bzcat` command prints the uncompressed form of a compressed file to `stdout`.

```
$ bzcat [options] filename
```

- To view the content of the `dante.bz2` compressed file, enter the following command:

```
$ bzcat dante.bz2 | less
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
... (output truncated)
```

Note: The `bzcat` command interprets the compressed data and displays the content of the file as if it were not compressed.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: The `bzcat` command does not change the content of the compressed file. The compressed file remains on the disk in compressed form.

Quiz



Which command has packaging and compression capabilities, in addition to archiving features?

- a. The `tar` command
- b. The `zip` command



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Quiz



The Oracle Solaris `gzcat` command is used for viewing files that have been compressed by using the `gzip` command.

- a. True
- b. False



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Quiz



What is the output of the `zip file7.zip file4 file12` command?

- a. An error message: The files must be compressed separately, one per `zip` command.
- b. `file7.zip`, `file4.zip`, and `file12.zip`: The compressed versions of each file
- c. `file7.zip`: The packaged and compressed zip file that contains two compressed files, `file4` and `file12`



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Practice 9: Overview

This practice covers the following topics:

- 9-1: Archiving and retrieving files
- 9-2: Compressing and restoring files
- 9-3: Performing remote connections and file transfers



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find these tasks for Practice 9 in your Activity Guide.

Agenda

- Archiving and retrieving files
- Compressing, viewing, and uncompressing files
- Performing remote connections and file transfers



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

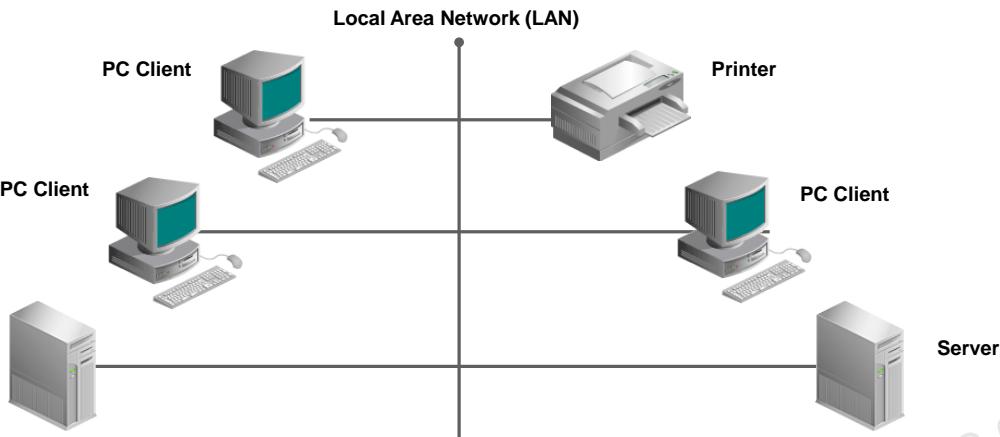
Computer Networking: Introduction

- A computer network is a group of computer components connected with each other by communication channels that allows sharing of resources and information.
- A computer system on a network is called a **host** which could be a Personal Computer (PC) client, a server, or a piece of network hardware such as a bridge, router, or a switch.
 - The **local host** is your current working system, usually a PC client or the server you are connected to.
 - A **remote host** is a different system, usually a server that you access from your local host.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Layout of a Basic Network



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The illustration depicts the relationship between the network and the host. The PC Client is a host or a process that uses services from another program, known as a server.

OpenSSH and Remote Network Connections

- OpenSSH (also known as OpenBSD Secure Shell) is a suite of network security utilities using the Secure Shell (SSH) network protocol.
- The utilities in the suite of OpenSSH packages provide:
 - sshd (a secure shell server daemon), which provides a secure end-to-end encrypted connection in an unsecure network
 - ssh (secure shell), which connects a client to a server
 - scp (secure copy), which copies files securely
 - sftp (secure ftp), which provides a secure file transfer protocol connection
- Remote login network connections can occur between a *client* machine and a *server* running sshd and between one *server* to another *server* running sshd.
- Each new connection/session is authenticated with a username and password.
- Once the session is authenticated and established, both the local and remote hosts communicate with each other via the Secure Shell (SSH) network protocol.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

In addition, to the 4 commands above OpenSSH also has a ssh-keygen command to generate public and private key pairs for authentication when connecting in Oracle's Cloud Environment.

```
$ ssh-keygen -t [ rsa1 | dsa | ecdsa | rsa ]
```

The type of key pairs being generated depends on the version of OpenSSH: “rsa1” Version 1 and “dsa”, “ecdsa” or “rsa” Version 2.

When prompted enter a passphrase (the longer the passphrase the more secure).

Using Secure Shell (ssh) for Remote Login

- The SSH network protocol provides a secure encrypted communication between two untrusted hosts over an unsecure network.
- The `ssh` (secure shell) client command allows you to connect and log in to a specified remote host.

```
$ ssh [options] [-l login_name | username@]hostname [command]
```

- `ssh` can use public-key encryption to authenticate a remote login session.
 - In public-key encryption, the `ssh-keygen` command generates a public-key that can be copied to all hosts that intend to communicate with the holder of the matching private-key.
- For more information about the `ssh` command options, see the `ssh` man page.

Note: In Oracle Solaris and Oracle Linux, OpenSSH is installed by default and is usable.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

A common option in `ssh` is to use a “-X” to enable X11 forwarding.

```
$ ssh -X oracle@s11-server1
```

Note: The best practice is to set SSH to not permitted for root user because any user would be able to `su` – into root. This provides a layer of protection for the root user.

Copying Files and Directories Between a Local and Remote Host: scp Command

- The `scp` (secure copy) command securely copies files and directories both ways between a local and a remote host.
- To copy files from a local directory to a remote host, use the following command syntax:

```
$ scp [options] SourceFile [username@]hostname:/directory/TargetFile
```

- To copy the `dante` file from the local directory to the `/tmp` directory on a remote system called `host2`, as the logged in user, enter the following command:

```
$ scp dante host2:/tmp
```

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

Note: The `[username@]` syntax is needed only when connecting as a different user other than the logged in user and requires that you know that user's password.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Caution: Do not use the `/tmp` directory for long-term storage. The `/tmp` directory in Oracle Solaris is emptied each time you reboot the system. On Oracle Linux, the System Administrator often runs a script to periodically empty out the `/tmp` directory.

Reversing the Direction and Copying Files from a Remote Host to a Local Host

- To copy files from a remote host to a local directory, use the following command syntax:

```
$ scp [username@]hostname:/directory/SourceFile TargetFile
```

- To copy the `dante` file from a remote host called `host2` to the local `/tmp` directory, as the logged in user, enter the following command:

```
$ scp host2:/tmp/dante /tmp
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: The `SourceFile` file is your original file and the `TargetFile` file is the copy of the original file.

Copying Local Directories to and from a Remote Host

- The `scp` command with the `-r` option recursively copies entire directories to and from another system.
- To copy the `perm` subdirectory in the local home directory to the `/tmp` directory on the remote system called `host2`, as the logged in user, enter the following command:

```
$ scp -r ~/lab/perm host2:/tmp
```

- To reverse the direction of the copy from the remote host to a local directory:

```
$ scp -r host2:/tmp ~/lab/perm
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

If your current working directory contains the file or directory that you want to copy, use the file or directory name. You do not need to use the absolute path name.

Quiz



Identify the correct command to copy the /opt/dante file from a remote host to the /tmp directory in your system.

- a. scp [username@]host2:/dante/tmp
- b. dante scp [username@]host2:/tmp
- c. [username@]host2/scp dante:/tmp
- d. scp [username@]host2:/opt/dante /tmp



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

File Transfer Protocol (FTP): Introduction

- The File Transfer Protocol (FTP) is a network protocol used for exchanging files over a TCP/IP network, not to be confused with the `ftp` userspace command.
- FTP implements user-based password authentication.
- FTP also allows anonymous user access, where the password is usually a valid email address.
- You can access a remote server for exchanging files securely using the `sftp` command:

```
$ sftp [options] [username@]hostname
```

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

Note: Although Oracle Solaris has both the `ftp` and the `sftp` client software, in this course, we will use only the more secure `sftp` client software found on Oracle Linux.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

When you access a remote system using the `sftp` command, some file and directory access commands, such as the `ls` and `cd` commands, are available at the `sftp>` prompt. Refer to either the `sftp` man page for additional information about `sftp` commands or use `sftp -help`.

Using OpenSSH's `sftp` to Transfer Files Either Remotely or Locally

- The `sftp` (secure `ftp`) command is an interactive file transfer program and performs all operations, such as file access, transfer, and management over an encrypted SSH transport.
- Being an extension of the OpenSSH protocol, `sftp` can use many of the features of SSH, such as public key authentication and encryption to enforce security.
- `sftp` only uses the `binary` transfer mode which is a byte-for-byte transfer mode.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Before Using sftp to Transfer Files, One Needs to Know the File Type and the Destination to Which the File Is Being Transferred

- In the lesson titled “Working with Files and Directories,” you learned how to determine a file’s file type by using the `file` command.

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

- Although dante’s file type is ASCII English text, that *file type* has different characteristics on different operating systems.
- The symbol that is used to represent the end-of-line or newline varies from:
 - Mac pre OS X, which uses CR (a single character--carriage return) to
 - Mac OS X, which uses the same ^M used by UNIX and Linux to
 - Microsoft, which uses CRLF (a double character--carriage return plus line feed) to
 - UNIX and Linux, which uses ^M (a single character--Ctrl-M)
- Before a file can be transferred, its format may need to be converted based on the destination to which the file is being transferred.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Using dos2unix or unix2dos Commands to Convert the File's Format Based on the Destination to Which the File is Being Transferred

- If the file is being transferred from Microsoft to UNIX, Linux, or MAC, then you need to use the `dos2unix` command:

```
$ dos2unix dante
dos2unix: converting the file dante to Unix format ...
```

- If the file is being transferred from UNIX, Linux, or MAC to Microsoft, then you need to use the `unix2dos` command:

```
$ unix2dos dante
unix2dos: converting the file dante to DOS format ...
```

- As the file is being converted to a new format, the original file is overwritten.
- For more information about the `dos2unix` and `unix2dos` command options, see the `dos2unix` man page.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: In Oracle Solaris, the full contents of the file are displayed as output, instead of the message you see in the slide for Oracle Linux.

Using the sftp Commands

The following are some of the frequently used sftp commands:

- **open**: Opens a connection to another computer on the network
- **get**: Transfers a file from the remote system to the local system's current directory
- **put**: Transfers a file from the local system to a directory on the remote system
- **mget**: Transfers multiple files from the remote system to the local system's current directory
- **mput**: Transfers multiple files from the local system to a directory on the remote system
- **bye, exit, and quit**: Quit or close the sftp environment



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following are some additional sftp commands:

- **help or ?:** Requests a list of all available sftp commands
- **cd pathname**: Changes directory path on the remote machine
- **lcd pathname**: Changes directory on your local machine
- **ls [options] pathname**: Lists the names of the files in the current remote directory
- **lls [options] pathname**: Lists the names of the files in the current local directory
- **mkdir pathname**: Makes a new directory in the current remote directory
- **lmkdir pathname**: Makes a new directory in the current local directory
- **pwd**: Finds out the pathname of the current directory on the remote machine
- **lpwd**: Finds out the pathname of the current directory on the local machine
- **rmdir pathname**: Removes or deletes a directory in the current remote directory
- **lrmdir pathname**: Removes or deletes a directory in the current local directory and others

Transferring Files Using sftp

```
[oracle@ol7-server1 ~] $ sftp s11-server
Password:
Connected to s11-server1.
sftp> pwd
Remote working directory: /home/oracle
sftp> ls lab/dante
lab/dante
sftp> get lab/dante
Fetching /home/oracle/lab/dante dante
/home/oracle/lab/dante                               100%  1319      1.3KB/s  00:00
sftp> lpwd
/home/oracle
sftp> lls
dante   Desktop   Downloads   Public    lab      Music   ...
sftp> bye
[oracle@ol7-server ~] $
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The objective is to connect to a remote server (s11-server1) as the `oracle` user and then use the `sftp` command to get a single file '`dante`' from the `~/lab` directory.

Transferring Multiple Files Using sftp

```
[oracle@s11-server1:~] $ sftp ol711-server
Password:
Connected to ol7-server1...
oracle@ol7-server1's password:
sftp> pwd
Remote working directory: /home/oracle
sftp> ls lab/fil*
lab/file.1      lab/file.2      lab/file.3      lab/file.4      lab/file1
lab/file2      lab/file3      lab/file4
sftp> get lab/fil*
Fetching /home/oracle/lab/file.1  file.1
Fetching /home/oracle/lab/file.2  file.2
Fetching /home/oracle/lab/file.3  file.3
Fetching /home/oracle/lab/file.4  file.4
Fetching /home/oracle/lab/file1  file1
/home/oracle/lab/file1      100%  1610      1.6KB/s      00:00

(continued next slide)
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The objective is to connect to a remote server (ol7-server1) as the oracle user and use the sftp command mget to get multiple files fil* from the ~/lab directory.

Transferring Multiple Files Using sftp

```
Fetching /home/oracle/lab/file2  file2
/home/oracle/lab/file2          100%   105   0.1KB/s      00:00
Fetching /home/oracle/lab/file3  file3
/home/oracle/lab/file3          100%   218   0.2KB/s      00:00
Fetching /home/oracle/lab/file4  file4
/home/oracle/lab/file4          100%   137   0.1KB/s      00:00
sftp> lpwd
/home/oracle
sftp> ll
dante  Documents  Desktop  Downloads  file.1    file.2
file.3  file.4    file1     file2     file4    lab ...
sftp> bye
[oracle@s11-server1:~] $
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Note: There is very little difference between the s11-server1 and o17-server1 output from the sftp command.

Quiz



Which is the most secure command for remotely logging in to another system within the network?

- a. rsh
- b. ssh
- c. telnet
- d. ftp



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Quiz



Select the three correct `sftp` command syntaxes to end an FTP session.

- a. `sftp> exit`
- b. `sftp> quit`
- c. `sftp> close`
- d. `sftp> bye`



ORACLE®

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Summary

In this lesson, you should have learned how to:

- Archive and retrieve files
- Compress, view, and uncompress files
- Perform remote connections and file transfers



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Practice 9: Overview

This practice covers the following topics:

- 9-1: Archiving and retrieving files
- 9-2: Compressing and restoring files
- 9-3: Performing remote connections and file transfers



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You will find this task for Practice 9 in your Activity Guide.

Oracle Cloud Computing

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Lesson Objectives

After completing this lesson, you should be able to:

- Describe Infrastructure as a Service (IaaS)
- Describe Oracle Private Cloud Appliance
- Describe Oracle OpenStack
- Describe Oracle Cloud Infrastructure
- Describe Key Concepts and Terms Used in Oracle Cloud Infrastructure
- Describe Oracle-Provided Images and Available Shapes in Oracle Cloud Infrastructure
- Describe the Task Flow to Launch an Oracle Cloud Infrastructure Instance
- Create an Oracle Cloud Infrastructure Virtual Cloud Network and Subnet
- Launch an Oracle Cloud Infrastructure Instance
- Attach an Oracle Cloud Infrastructure Block Storage Volume to an Instance



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

There are many Oracle cloud solutions and combinations to suit different customers needs:

- A Private Cloud solution, for example: Oracle Private Cloud Appliance, or building a solution with Oracle OpenStack
- An Oracle Cloud Infrastructure solution
- A hybrid solution

This lesson provides an introduction to each of these Oracle cloud solutions.

What Is Infrastructure as a Service (IaaS)?

- It is a form of cloud computing that provides infrastructure services over the Internet.
- It is required by all applications, databases, and middleware deployments.
- Users can access:
 - Computer processors
 - Storage
 - Networks
 - Other infrastructure resources
- IaaS is one of three “service models” in cloud computing. The other two are:
 - SaaS – Software as a Service
 - PaaS – Platform as a Service



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Infrastructure as a Service (IaaS) is a form of cloud computing that provides infrastructure services over the Internet. One of the main features of IaaS is that users can access their applications and data from anywhere, as long as they can access the Internet. IaaS provides the following benefits:

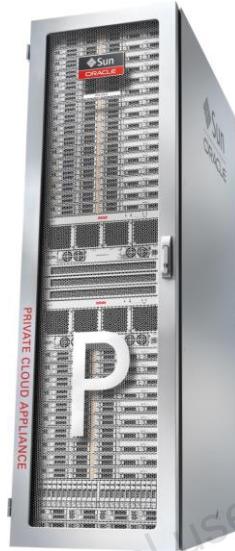
- The user has access to the foundation level of the computing “task”—compute capacity, network bandwidth, and storage capacity.
- The IaaS provider is responsible for providing the hardware, maintenance, and cloud infrastructure up to the virtualization level.
- The user can create virtual machines as required and decide what operating system and/or apps to use.

The other two service models in cloud computing are:

- **Software as a Service (SaaS):** The user has access only to the applications provided by the SaaS provider. The provider has full control of the entire infrastructure including applications.
 - Example: An email provider
- **Platform as a Service (PaaS):** The user is able to deploy applications using programs and tools provided by the provider. The provider has full control of the infrastructure up to the point where users are able to modify their applications and environment.
 - Example: A web hosting provider that offers tools and programs to deploy a website

What Is Oracle Private Cloud Appliance?

- It is preconfigured for stability, high availability, and automation.
- Orchestration software automatically handles new server hardware.
- It supports provisioning of Infrastructure and Platform as a Service (IaaS and PaaS) on demand.
- Private Cloud Appliance Virtualization
 - Focus on services, not managing hardware
 - Created to handle demanding workloads
 - Unified support for the software solution stack
- The Private Cloud Appliance simplifies application deployment and management.



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle Private Cloud Appliance, shown in the slide, is a converged infrastructure appliance. Because the servers, switches, and storage are preconfigured, Private Cloud Appliance is ready to run almost immediately after installation.

The hardware is preconfigured at the factory for high availability and for automation. This minimizes the risks of misconfiguration later on. Insert additional servers and attach them to the preconfigured cables to connect them to the network. Orchestration software automatically detects and integrates new servers to increase processor and memory capacity for virtualization.

The Private Cloud Appliance environment supports the provisioning of Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) among other services. With templates and assemblies, you can rapidly meet demands for applications without having to consider additional network and storage capacity, or cabling and power, to host them.

Virtualization enables you to provide application platforms and infrastructure as services. You manage policies to provide services on demand, on a schedule, and for a particular length of time. You manage hardware to maintain the virtualization environment and not in response to application demands.

The Private Cloud Appliance virtualization environment has been created with significant capacity for demanding workloads. Private Cloud Appliance is built with Oracle hardware and software. Because Private Cloud Appliance is a unified solution, Oracle supports the entire system and its software. This approach separates the concerns of managing hardware in the data center from the demands of rapidly changing applications. The result is simplified application deployment and management. For more information about Private Cloud Appliance, go to <http://www.oracle.com/servers/private-cloud-appliance> or take the *Oracle Private Cloud Appliance Administration* course at <http://www.oracle.com/servers/private-cloud-appliance>.

What Is Oracle OpenStack?

- Oracle OpenStack is a cloud management software for managing large pools of compute, storage, and networking resources.
- Based on an OpenStack community release, Oracle OpenStack is an enterprise-grade solution for managing an entire IT environment.
- It rapidly deploys Oracle and third-party applications across shared compute, network, and storage resources.
- It accelerates application deployment with self-service VM creation.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle OpenStack offers the ability to deploy different deployment configurations depending on the demand. As an IaaS cloud management platform, it centralizes enterprises' cloud operations for OpenStack operators to manage and deploy resources within an IT network environment.

Oracle OpenStack is a software-defined network (SDN) solution in deploying VMs and virtual networks abstracted from the physical infrastructure of a data center. Optimized for Oracle Linux, Oracle OpenStack securely streamlines the deployment of compute resources while decreasing the time to deploy applications.

To learn more go to <http://www.oracle.com/openstack> or take the *Oracle OpenStack for Oracle Linux Getting Started* course at <http://oracle.com/education/linux>.

What Are Oracle Cloud Infrastructure Services?

Set of complementary cloud services that enable you to build and run a wide range of applications and services in a highly-available hosted environment.

- Compute Service
 - Provision and manage bare metal compute instances or virtual machine instances.
- Networking Service
 - Create and manage the network components for your cloud resources.
- Block Volume Service
 - Provides high-performance network storage capacity.
- Database Service
 - Build, scale, and secure Oracle databases with license-included pricing.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle Cloud Infrastructure is a set of complementary cloud services that enable you to build and run a wide range of applications and services in a highly-available hosted environment. Oracle Cloud Infrastructure offers high-performance compute capabilities (as physical hardware instances) and storage capacity in a flexible overlay virtual network that is securely accessible from your on-premises network.

Use the **Compute Service** to provision and manage bare metal compute instances. You can launch an Oracle bare metal compute resource in minutes. Provision instances as needed to deploy and run your applications, just as you would in your on-premises data center. Managed Virtual Machine (VM) instances are also available for workloads that don't require dedicated physical servers or the high-performance of bare metal instances.

Use the **Networking Service** to create and manage the network components for your cloud resources. You can configure your virtual cloud network (VCN) with access rules and gateways to support routing of public and private internet traffic.

The **Block Volume Service** provides high-performance network storage capacity that supports a broad range of I/O intensive workloads. You can use block volumes to expand the storage capacity of your compute instances, to provide durable and persistent data storage that can be migrated across compute instances, and to host large databases.

The **Database Service** lets you easily build, scale, and secure Oracle databases with license-included pricing in your Oracle Cloud Infrastructure cloud. You create databases on DB Systems, which are bare metal servers with local NVMe flash storage. You launch a DB System the same way you do a bare metal instance, you just add some additional configuration parameters. You can then use your existing tools, RMAN, and the database CLI to manage your databases in the cloud the same way you manage them on-premises.

What Are Oracle Cloud Infrastructure Services?

- Identity and Access Management (IAM) Service
 - Control access to Oracle Cloud Infrastructure Services.
- Load Balancing Service
 - Create a highly available load balancer within your virtual cloud network (VCN).
- Object Storage Service
 - Provides high throughput storage for unstructured data.
- Audit Service
 - Audit log events for security audits, to track usage, and to help ensure compliance.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Use the **Identity and Access Management (IAM) Service** to control access to Oracle Cloud Infrastructure. Create and manage compartments, users, groups, and the policies that define permissions on resources.

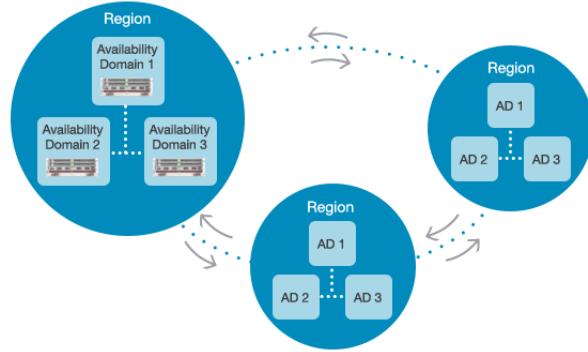
The **Load Balancing Service** allows you to create a highly available load balancer within your virtual cloud network (VCN) so that you can distribute internet traffic to your compute instances within the VCN.

The **Object Storage Service** provides high throughput storage for unstructured data. Object storage enables near infinite storage capacity for large amounts of analytic data, or rich content like images and videos. Block volumes can be backed up to the highly durable Object Storage Service for added durability.

The **Audit Service** provides visibility into activities related to your Oracle Cloud Infrastructure resources and tenancy. Audit log events can be used for security audits, to track usage of and changes to Oracle Cloud Infrastructure resources, and to help ensure compliance with standards or regulations.

Key Concepts and Terms Used in Oracle Cloud Infrastructure

- Regions and Availability Domains
 - Bare metal services are physically hosted in Regions and Availability Domains.
 - A Region is a localized geographic area composed of several Availability Domains.
 - An Availability Domain is one or more data centers located within a region.
 - Availability Domains are isolated from each other, fault tolerant, and very unlikely to fail simultaneously.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following lists and describes the key concepts and terms used in Oracle Cloud Infrastructure.

Regions and Availability Domains

Oracle Cloud Infrastructure are physically hosted in regions and Availability Domains. A region is a localized geographic area, and an Availability Domain is one or more data centers located within a region. A region is composed of several Availability Domains. Bare metal resources are either region-specific, such as a Virtual Cloud Network, or Availability Domain-specific, such as a compute instance.

Availability Domains are isolated from each other, fault tolerant, and very unlikely to fail simultaneously or be impacted by the failure of another Availability Domain. When you configure your cloud services, use multiple Availability Domains to ensure high availability and to protect against resource failure. Be aware that some resources must be created within the same Availability Domain, such as an instance and the storage volume attached to it.

Key Concepts and Terms

- Tenancy
 - A Tenancy is a secure and isolated partition within Oracle Cloud Infrastructure where you can create, organize, and administer your cloud resources.
 - Oracle creates a Tenancy for your company when you sign up for Oracle Cloud Infrastructure.
 - Your Tenancy is the root Compartment that holds all your cloud resources.
- Compartments
 - Compartments allow you to organize and control access to your Oracle Cloud Infrastructure resources.
 - Although you can see the list of all compartments, you must be granted permissions on a compartment to see or take action on its resources.
 - You can create additional Compartments within the Tenancy (root compartment).
 - When you create a Oracle Cloud Infrastructure resource, you must specify which compartment you want the resource to belong.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The lists of the key concepts and terms used in Oracle Cloud Infrastructure continues.

Tenancy

A tenancy is a secure and isolated partition within Oracle Cloud Infrastructure where you can create, organize, and administer your cloud resources. When you sign up for Oracle Cloud Infrastructure, Oracle creates a tenancy for your company. A tenancy can be subscribed to multiple regions.

Compartments

Compartments allow you to organize and control access to your cloud resources. A compartment is a collection of related resources (such as instances, virtual cloud networks, block volumes) that can be accessed only by certain groups that have been given permission by an administrator. A compartment is to be thought of as a logical group, rather than a physical container. When you begin working with resources in the Console, the compartment acts as a filter for what you are viewing.

Your tenancy is the root compartment that holds all your cloud resources. You then create additional compartments within the tenancy (root compartment) and corresponding policies to control access to the resources in each compartment. When you create a cloud resource such as an instance, block volume, or cloud network, you must specify to which compartment you want the resource to belong.

The correct use of compartments ensures that each user has access to only the resources they need.

Key Concepts and Terms

The screenshot shows the Oracle Cloud Infrastructure Identity console. The top navigation bar includes the Oracle logo, TENANCY (gsebmcs00002), REGION (us-phoenix-1), and user information (demo.user38). The main menu on the left is set to 'Identity' and includes options like 'Users', 'Groups', 'Policies', 'Compartments' (which is selected and highlighted in blue), and 'Federation'. The central area is titled 'Compartments' and shows a list of existing compartments. A 'Create Compartment' button is located at the top of the list. The first compartment listed is 'RC' (root), with a description 'The root Compartment of the tenancy', OCID '...3f7cyq', and created on 'Thu, 01 Dec 2016 09:48:36 GMT'. The second compartment is '16' (OCID '...pyue7q'), created on 'Thu, 01 Dec 2016 09:48:36 GMT'. The third compartment is '46' (OCID '...hmqlaq'), created on 'Thu, 01 Dec 2016 09:49:40 GMT'. Each compartment row includes a 'Rename Compartment' button.

The lists of the key concepts and terms used in Oracle Cloud Infrastructure continues.

Console

The console is a web-based user interface you can use to access and manage Oracle Cloud Infrastructure. The screen shows the console.

The Tenancy on the slide, which is selected when you log in, is gsebmcs00002.

The Region on the slide is us-phoenix-1. You need to select a region if your tenancy is subscribed to multiple regions.

The screen shows a selection of the existing Compartments, along with the option to create additional compartments. The root Compartment is shown first on this screen. When you create a cloud resource such as an instance, block volume, or cloud network, you must specify to which compartment you want the resource to belong.

The Oracle Cloud Infrastructure services are offered through the drop-down menus on the top-right corner of the console. The options correspond to the services as follows:

- **Identity:** Identity and Access Management (IAM) Service
- **Compute:** Compute Service
- **Database:** Database Service
- **Networking:** Networking Service and Load Balancing Service
- **Storage:** Block Volume Service and Object Storage Service
- **Audit:** Audit Service

You also have access to Support (My Oracle Support) and online Documentation from the console.

Key Concepts and Terms

- Instance
 - An instance is a compute host running in the cloud.
 - Bare metal compute instances run on bare metal servers without a hypervisor.
 - You maintain sole control of the physical CPU, memory, and Network Interface Card.
 - You do not share the physical machine with any other tenants.
 - Managed Virtual Machine (VM) instances are also available for workloads that don't require dedicated physical servers or the high-performance of bare metal instances.
- Image
 - The image is a template of a virtual hard drive that defines the operating system and other software for an instance, for example Oracle Linux.
 - When you launch an instance, you define its characteristics by choosing its image.
- Shape
 - The shape specifies the number of CPUs and amount of memory allocated to the instance, and specifies if the instance is a bare metal instance or a VM instance.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The lists of the key concepts and terms used in Oracle Cloud Infrastructure continues.

Instance

An instance is a compute host running in the cloud. An Oracle Cloud Infrastructure compute instance allows you to utilize hosted physical hardware, as opposed to the traditional software-based virtual machines, ensuring a high level of security and performance.

Image

The image is a template of a virtual hard drive that defines the operating system and other software for an instance, for example Oracle Linux. When you launch an instance, you can define its characteristics by choosing its image. Oracle provides a set of images you can use. You can also save an image from an instance that you have already configured to use as a template to launch more instances with the same software and customizations.

Shape

In the Compute Service, the shape specifies the number of CPUs and amount of memory allocated to the instance. The shape also specifies if the instance is a bare metal instance or a virtual machine instance.

Oracle-Provided Images and Available Shapes

Image	Name	Description
Oracle Linux 7 Unbreakable Enterprise Kernel Release 4	Oracle-Linux-7.x-<date>-<number>	The Unbreakable Enterprise Kernel (UEK) is Oracle's optimized operating system kernel for demanding Oracle workloads. X7 shapes are supported in the latest image version. For more information, see Oracle-Provided Image Release Notes .
Oracle Linux 6 Unbreakable Enterprise Kernel Release 4	Oracle-Linux-6.x-<date>-<number>	The Unbreakable Enterprise Kernel (UEK) is Oracle's optimized operating system kernel for demanding Oracle workloads. X7 shapes are not supported with this image.
CentOS 7	CentOS-7-<date>-<number>	CentOS is a free, open-source Linux distribution suitable for use in enterprise cloud environments. For more information, see https://www.centos.org/ .
CentOS 6	CentOS-6.x-<date>-<number>	CentOS is a free, open-source Linux distribution that is suitable for use in enterprise cloud environments. For more information, see https://www.centos.org/ . X7 shapes are not supported with this image.

Bare Metal Shapes

Shape	Instance Type	OCPUs	Memory (GB)	Local Disk (TB)	Network Bandwidth*	Maximum VNICs Total**
BM.Standard1.36	Standard compute capacity	36	256	Block storage only	10 Gbps	16
BM.HighIO1.36	High I/O compute capacity	36	512	12.8TB NVMe SSD	10 Gbps	16

VM Shapes

VMs are an option that provides flexibility in compute power, memory capability, and network resources for lighter applications. You can use Block Volume to add network-attached block storage as needed.

Shape	OCPUs	Memory (GB)	Local Disk (TB)	Network Bandwidth*	Maximum VNICs Total**
VM.Standard1.1	1	7	Block Storage only	Up to 600 Mbps	2
VM.Standard1.2	2	14	Block Storage only	Up to 1.2 Gbps	2



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

This slide shows some of the available Oracle-provided images and shapes you can select when creating an Oracle Cloud Infrastructure instance.

For more images, see: <https://docs.us-phoenix-1.oraclecloud.com/Content/Compute/References/images.htm>

For more shapes, see: <https://docs.us-phoenix-1.oraclecloud.com/Content/GSG/Concepts/concepts.htm>

Key Concepts and Terms

- Virtual Cloud Network (VCN)
 - A VCN is a virtual version of a traditional network on which your instances run.
 - VCN includes subnets, route tables, and gateways.
 - VCN resides within a region but subnets can belong to different Availability Domains.
 - VCN can have an optional Internet Gateway to handle public traffic.
 - VCN can have an optional IPSec VPN connection to securely extend your on-premises network.
- Block Volume
 - A virtual disk that provides persistent block storage space for instances.
 - You can move block volumes from one instance to another without loss of data.
- Object Storage
 - A storage architecture that allow you to store and manage data as objects.
 - Data files can be of any type and up to 50 GB in size.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The lists of the key concepts and terms used in Oracle Cloud Infrastructure continues.

Virtual Cloud Network (VCN)

A Virtual Cloud Network is a virtual version of a traditional network—including subnets, route tables, and gateways—on which your instances run. A cloud network resides within a single region but can cross multiple Availability Domains. You can define subnets for a cloud network in different Availability Domains, but the subnet itself must belong to a single Availability Domain. You need to set up at least one cloud network before you can launch instances. You can configure the cloud network with an optional Internet Gateway to handle public traffic, and an optional IPSec VPN connection to securely extend your on-premises network.

Block Volume

A block volume is a virtual disk that provides persistent block storage space for Oracle Cloud Infrastructure instances. Use a block volume just as you would a physical hard drive on your computer, for example, to store data and applications. You can detach a volume from one instance and attach it to another instance without loss of data.

Object Storage

Object Storage is a storage architecture that allows you to store and manage data as objects. Data files can be of any type and up to 50 GB in size. After you upload data to the Object Storage Service it can be accessed from anywhere. Use the Object Storage Service when you want to store a very large amount of data that does not change very frequently. Some typical use cases for the Object Storage Service include data backup, file sharing, and storing unstructured data like logs and sensor-generated data.

Task Flow to Launch an Oracle Cloud Infrastructure Instance

- Create an SSH key pair.
- Create or choose a compartment for your resources.
- Create or choose a VCN (virtual cloud network).
- Create a Subnet for the VCN.
- Launch an instance.
- Connect to your instance.
- Provision and manage block storage volumes (optional)
 - Add a block storage volume
 - Attach the volume to an instance
 - Connect a volume to an instance's guest OS



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Instances use an SSH key pair instead of a password to authenticate a remote user. If you do not already have a key pair, your first task is to create one. You can use PuTTYgen on Windows or use the `ssh-keygen` utility on Oracle Linux to create an SSH key pair. You can have multiple key pairs to use for different instances, or you can use the same key pair for all instances.

Next, choose a compartment to hold your cloud resources. Your tenancy is the root compartment but you can create additional compartments within the tenancy (root compartment) and corresponding policies to control access to the resources in each compartment. When you create a cloud resource such as an instance, block volume, or cloud network, you must specify to which compartment you want the resource to belong.

Next, create a VCN (virtual cloud network) with subnets. You need to set up at least one cloud network before you can launch instances.

Next, launch the instance into one of the subnets for the VCN. You need to provide an instance name, specify an Availability Domain, and select an image, shape, and build. You also select the VCN desired, along with a subnet. A box is checked by default to assign a public IP address, with a private IP address being optional. You can specify an optional hostname and an FQDN is generated automatically. You must specify the public SSH key(s) from your generated key pair(s) by either choosing the SSH key file(s) from your computer, or pasting the public key(s) from your key pair(s) into the form. You can then click the **Launch Instance** button. Launching an Instance takes several minutes as its status moves from PROVISIONING to RUNNING. You then need to wait another minute for the operating system to boot before you can SSH to the Instance to connect to it.

Lastly, you can optionally provision block storage volumes for instances to meet your storage and application needs.

Setting up a Virtual Cloud Network (VCN)

The screenshot shows the 'Create Virtual Cloud Network' dialog box. It includes fields for 'CREATE IN COMPARTMENT' (set to 'c13'), 'NAME OPTIONAL' (set to 'VCN for 192.168'), and 'CIDR BLOCK' (set to '192.168.0.0/16'). Under 'DNS RESOLUTION', the 'USE DNS HOSTNAMES IN THIS VCN' checkbox is selected. The 'DNS LABEL' field contains 'vcnfor192168'. The 'DNS DOMAIN NAME (READ-ONLY)' field shows 'vcnfor192168.oraclevcn.com'. At the bottom, there is a 'Create Virtual Cloud Network' button and a checked 'View detail page after this resource is created' checkbox.

To display this screen:

1. Click Networking
2. Click Virtual Cloud Networks
3. Click Create Virtual Cloud Network

Provide parameters for the VCN

Click Create Virtual Cloud Network button to save

Create one or more Subnets for the VCN

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

When you work with Oracle Cloud Infrastructure, one of the first steps is to set up a Virtual Cloud Network (VCN) for your cloud resources.

To create a VCN, from the Oracle Cloud Infrastructure console, click the Networking menu option and select an appropriate compartment from the drop down list. You can then click the Create Virtual Cloud Network button. The Create Virtual Cloud Network window is shown on the slide.

For the cloud network you must specify a single, contiguous IPv4 Classless Inter-Domain Routing (CIDR) block. Oracle recommends the private IP address ranges specified in RFC 1918 (10.0.0.0/8, 172.16/12, and 192.168/16). Example: 10.0.0.0/16. The cloud network can range from /16 to /30 and must not overlap with your on-premise network or another VCN you peer with (VCN peering = connecting multiple VCNs). It is possible to use a publicly routable range. You cannot change the size of the cloud network after creation.

You can optionally assign a friendly name to the cloud network. It does not have to be unique, and you can change it later (with the API, not the Console). Oracle automatically assigns the cloud network a unique identifier called an Oracle Cloud ID (OCID).

If you want the instances in the VCN to have DNS hostnames (which can be used with the Internet and VCN Resolver, a built-in DNS capability in the VCN), select the check box for Use DNS Hostnames in this VCN (it is selected by default). Then you can specify a DNS label for the VCN, or the Console generates one for you. The dialog box automatically displays the corresponding DNS Domain Name for the VCN (<VCN_DNS_label>.oraclevcn.com).

Click Create Virtual Cloud Network. The cloud network is then created and displayed on the Virtual Cloud Networks page in the compartment you chose. To delete a cloud network, it must be empty and have no attached gateways. Next, create one or more subnets in the cloud network.

Working with VCN Subnets

To display this screen:

1. Click Networking
2. Click Virtual Cloud Networks
3. Click an existing VCN from list
4. Click Create Subnet

Provide parameters for the Subnet

Click Create button to save

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

A subnet is a subdivision of a cloud network and each instance resides in a subnet. Subnets contain virtual network interface cards (VNICs), which attach to instances. Each subnet exists in a single Availability Domain (which must be specified) and consists of a contiguous range of IP addresses that do not overlap with other subnets in the cloud network (a CIDR Block must be specified). Example: 192.168.0.0/24. You cannot change the size of the subnet after creation. Subnets can be either public or private. You choose this during subnet creation, and you cannot change it later. The subnet acts as a unit of configuration, such that all instances in a given subnet use the same route table, security lists, and DHCP options.

Optionally assign a friendly name to the subnet. Oracle automatically assigns the subnet a unique identifier called an Oracle Cloud ID (OCID). You can also add a DNS label for the subnet, which is required if you want the subnet's instances to use the Internet and VCN Resolver feature for DNS. This option is available only if you provided a DNS label for the VCN during creation.

You can optionally specify a route table for the subnet. If you do not, the cloud network's default route table is associated with the subnet. After creating the subnet, you cannot change which route table is associated with it, but you can change the route rules in the table.

You can optionally specify from one to five security lists for the subnet. If you do not specify any, the cloud network's default security list is associated with the subnet. After creating the subnet, you cannot change which security lists are associated with it, but you can change the rules in the lists. Security list rules are enforced at the instance level, even though the list is associated at the subnet level. You can also optionally associate a set of DHCP options with the subnet during creation. All instances in the subnet receive the configuration specified in that set of DHCP options. If you do not specify a set, the cloud network's set of default DHCP options is associated with the subnet.

If you enable compartment selection, you must specify the compartment where you want the subnet to reside, as well as for the Route Table and DHCP Options.

Launching an Instance

The screenshot shows the 'Launch Instance' wizard with the following configuration:

- NAME:** OracleLinux7/Update4-1
- AVAILABILITY DOMAIN:** GIAT-PHO-AD-3
- IMAGE SOURCE:** Oracle-Linux-7.4 (selected)
- IMAGE OPERATING SYSTEM:** Oracle Linux 7.4
- INSTANCE TYPE:** VIRTUAL MACHINE (selected)
- INSTANCE SHAPE:** VM.Standard1.4
- IMAGE BUILD:** 2017.11.15-0 (latest)
- VIRTUAL CLOUD NETWORK:** VCN for 192.168
- SUBNET:** Subnet 1 for VCN 192.168
- PRIVATE IP ADDRESS:** (None)
- HOSTNAME:** (None) - set to instance1
- FULLY QUALIFIED DOMAIN NAME:** instance1.subnet1.vcnfor192.168.oraclevcn.com
- SSH KEYS:** (None) - choose to paste SSH key files

ORACLE

To display this screen:

1. Click Compute
2. Click Instances
3. Click Launch Instance

Provide parameters for the Instance

Click Launch Instance button (not shown) to save

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

When you launch an instance, it is automatically attached to a Virtual Network Interface Card (VNIC) in the cloud network's subnet and given a private IP address from the subnet's CIDR. You can either let the address be automatically assigned, or specify a particular address of your choice. The private IP address lets instances within the cloud network communicate with each other. They can instead use fully qualified domain names (FQDNs) if you've set up the cloud network for DNS.

You have the option to assign the instance a public IP address if the subnet is public. A public IP address is required to communicate with the instance over the Internet, and to establish a Secure Shell (SSH) or Remote Desktop Protocol (RDP) connection to the instance from outside the cloud network.

Before you launch an instance, be sure you have created a VCN and subnet, and you have an SSH key pair.

Enter the name for the instance. You can add or change the name later. An Oracle Cloud Identifier (OCID) uniquely identifies the instance. Select the Availability Domain in which you want to run the instance.

Select the image to use for booting the instance. The image determines the operating system and other software for the instance. You can optionally select an image build. Select the shape, which determines the number of CPUs, amount of memory, and other resources allocated to a newly created instance. You can choose either Bare Metal or VM shapes.

Select the Virtual Cloud Network (VCN) in which to launch the instance. Select the subnet within the cloud network to attach the instance to. The subnets are either public or private. Private means the instances in that subnet cannot have public IP addresses.

Paste or browse the SSH public key portion of the key pair you want to use for SSH access to the instance. Click the Launch Instance button (not shown on slide) to save the instance.

Viewing Instance Details

The screenshot shows the Oracle Cloud Infrastructure Instance Details page. At the top, it displays 'TENANCY' (asebmcs00001) and 'REGION' (us-phoenix-1). On the right, there are navigation links for Home, Identity, Compute, Database, Networking, Storage, and Audit, along with a user dropdown (demo.user13). Below the header, the instance name 'OracleLinux7Update4-1' is shown, along with a green thumbnail icon labeled 'RUNNING'. A row of buttons includes 'Create Custom Image' (gray), 'Start' (gray), 'Stop' (gray), 'Reboot' (gray), and 'Terminate' (red). The main content area is divided into sections: 'Instance Information' (Availability Domain: GxATPHX-AD-3, OCID: ..., Launched: Wed, 22 Nov 2017 22:44:31 GMT, Compartment: c13), 'Image' (Oracle-Linux-7.4-2017.11.15-0), 'Region' (phx), 'Shape' (VM.Standard1.4), and 'Virtual Cloud Network' (VCN for 192.168). Below this, 'Primary VNIC Information' is listed with Private IP Address (192.168.0.2), Public IP Address (129.146.99.236), Fully Qualified Domain Name (instance1...), and Subnet (Subnet 1 for VCN 192.168). A note at the bottom states: 'This Instance's traffic is controlled by its firewall rules in addition to the associated Subnet's Security Lists.' The bottom of the page features the Oracle logo and a copyright notice: 'Copyright © 2018, Oracle and/or its affiliates. All rights reserved.'

To view the details of an instance, from the Oracle Cloud Infrastructure console, click the Compute menu option, then click the Instances menu option, then click the instance name you want to view from the list of instances. The Instance Details window is shown on the slide.

From this window you can see Instance Information and Primary VNIC Information. The Public IP Address in this example is 129.146.99.236. You can use this IP address to access your instance from a remote system. For instances created from Oracle Linux images, the default user name is `opc`. To log on to an instance using `ssh`, specify the full path and file name containing the private key associated with the instance, along with the `opc` user name. For example, to log on with a private key contained in the file `id_rsa` in the `/root/.ssh` directory:

```
# ssh -i ~/.ssh/id_rsa opc@129.146.99.236
```

Some command output is shown after logging onto the Oracle Linux instance:

```
[opc@instance1 ~]$ hostname
instance1
[opc@instance1 ~]$ whoami
opc
[opc@instance1 ~]$ ls /home
opc
[opc@instance1 ~]$ uname -r
4.1.12-103.9.2.el7uek.x86_64
```

Additional command output is shown on the next page.

```
[opc@instance1 ~]$ cat /etc/oracle-release
Oracle Linux Server release 7.4
[opc@instance1 ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP qlen 1000
    link/ether 00:00:17:00:77:44 brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.2/24 brd 192.168.0.255 scope global dynamic ens3
            valid_lft 85162sec preferred_lft 85162sec
[opc@instance1 ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        14G    0   14G  0% /dev
tmpfs          14G    0   14G  0% /dev/shm
tmpfs          14G   17M  14G  1% /run
tmpfs          14G    0   14G  0% /sys/fs/cgroup
/dev/sda3       39G  1.8G  37G  5% /
/dev/sda1       512M  9.8M  502M  2% /boot/efi
tmpfs          2.8G    0   2.8G  0% /run/user/0
tmpfs          2.8G    0   2.8G  0% /run/user/1000
[opc@instance1 ~]$ sudo fdisk -l
WARNING: fdisk GPT support is currently new, and therefore in an experimental phase. Use at your own discretion.

Disk /dev/sda: 50.0 GB, 50010783744 bytes, 97677312 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disk label type: gpt
Disk identifier: 60FE6558-113D-430E-B828-C4C61A5B0DAD

#           Start            End         Size      Type             Name
 1          2048          1050623       512M  EFI System     EFI System
Partition
 2          1050624        17827839        8G  Linux swap
 3          17827840        97675263      38.1G Microsoft basic
```

Creating a Block Storage Volume

The screenshot shows the 'Create Block Volume' dialog box. It has fields for 'CREATE IN COMPARTMENT' (set to 'c13'), 'NAME' ('BlockVolume1'), 'AVAILABILITY DOMAIN' ('GxAT:PHX-AD-3'), and 'SIZE (IN GB)' ('50'). Below these fields is a note: 'Size must be between 50 GB and 16384 GB (16 TB). Volume performance varies with volume size.' At the bottom are two buttons: 'Create Block Volume' (highlighted in blue) and 'View detail page after this resource is created' (with a checked checkbox).

To display this screen:

1. Click Storage
2. Click Block Volumes
3. Click Create Block Volume

Provide parameters for the Volume

Click Create Block Volume to save



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The Oracle Cloud Infrastructure Block Volume Service lets you dynamically provision and manage block storage volumes. You can create, attach, connect and move volumes as needed to meet your storage and application requirements. When attached and connected to an instance, you can use a volume like a regular hard drive. Volumes can also be disconnected and attached to another instance without the loss of data.

To create a Block Storage Volume, from the Oracle Cloud Infrastructure console, click the Storage menu option, then click the Block Volumes menu option, then click Create Block Volume button. The Create Block Volume window is shown on the slide.

Choose a compartment to hold your block volume resource. Provide a user-friendly name or description. The Block Volume must be in the same Availability Domain as the instance. Choose the size of the block volume between 50 GB and 16384 GB (16 TB).

Click Create Block Volume to save. The volume is ready to attach to an instance when its icon no longer lists it as PROVISIONING, but AVAILABLE in the volume list.

Attaching a Block Storage Volume to an Instance

The screenshot shows a 'Attach Block Volume' dialog box. It has fields for 'BLOCK VOLUME COMPARTMENT' (set to 'c13') and 'BLOCK VOLUME' (set to 'BlockVolume1'). There is a checkbox for 'REQUIRE CHAP CREDENTIALS' which is not checked. A large blue 'Attach' button is at the bottom.

To display this screen:

1. Click Compute
2. Click Instances
3. Click the Name of the Instance to display the Instance Details
4. Click Attach Block Volume

Provide parameters for the Volume

Click Attach to save



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You can attach a volume to an instance in order to expand the available storage on the instance.

To attach a Block Storage Volume to an Instance, from the Oracle Cloud Infrastructure console, click the Compute menu option, then click the Instances menu option.

In the Instances list, locate the instance you want to attach to the volume to.

Click the Instance name to display the instance details.

Click Attach Block Volume to display the screen shown on the slide.

Select the volume you want from the Volume drop-down menu.

You can optionally enable CHAP (Challenge-Handshake Authentication Protocol). Block volumes on Oracle Cloud Infrastructure use iSCSI to connect to instances. CHAP is a security protocol used by iSCSI for authentication between a volume and an instance and is recommended by Oracle.

Click Attach.

When attached, you must still connect and mount the volume from the instance for the volume to be usable.

Connecting a Block Storage Volume to an Instance's Guest OS

iSCSI Commands & Information

ATTACH COMMANDS

```
sudo iscscliadm -m node -o new -T iqn.2015-12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789
sudo iscscliadm -m node -o update -T iqn.2015-12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789
sudo iscscliadm -m node -T iqn.2015-12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789
```

DETACH COMMANDS

```
sudo iscscliadm -m node -T iqn.2015-12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789
sudo iscscliadm -m node -o delete -T iqn.2015-12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789
```

IP ADDRESS AND PORT

169.254.2.2:3260

VOLUME IQN

iqn.2015-12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789

To display this screen:

1. Click Compute
2. Click Instances
3. Click the Name of the Instance to display the Instance Details
4. Scroll down to view the Attached Block Volumes
5. Click the Actions icon on your Block Volume's row
6. Click iSCSI Commands and Information menu option

Log on to your instance and run the three `iscscliadm` commands

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You use the iSCSI protocol to connect a Block Volume Service volume to an instance. After the volume is attached, you log on to the instance and use the `iscscliadm` command-line tool to configure the iSCSI connection. After you configure the volume, you can mount it and use it like a normal hard drive.

The Oracle Cloud Infrastructure console provides the iSCSI command information on the details page of the volume's attached instance. Click the Actions icon on your volume's row, and then click iSCSI Information.

Log on to your instance and issue the three `iscscliadm` commands.

Connecting a Block Storage Volume to an Instance's Guest OS

- Log on to your instance and issue the three `iscsiadm` commands:

```
[opc@instance1 ~]$ sudo iscsiadm -m node -o new -T ...
[opc@instance1 ~]$ sudo iscsiadm -m node -o update -T ...
[opc@instance1 ~]$ sudo iscsiadm -m node -T ...
```

- The `fdisk -l` command shows the newly attached disk, `/dev/sdb`, in this example:

```
[opc@instance1 ~]$ sudo fdisk -l |grep /dev
...
Disk /dev/sdb: 53.7 GB, 53687091200 bytes, 104857600 sectors
```

- You can then partition the new block device, create a file system on the device, create a mount point, and mount the new file system.
- Include the `_netdev` and `nofail` options on every non-root block volume in the `/etc/fstab` file.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The following set of Oracle Linux commands show the available block devices before and after issuing the `iscsiadm` commands. Before, there is only a single 50 GB disk, `/dev/sda`.

```
[opc@instance1 ~]$ sudo fdisk -l |grep /dev
WARNING: fdisk GPT support is currently new, and therefore in an
experimental phase. Use at your own discretion.
Disk /dev/sda: 50.0 GB, 50010783744 bytes, 97677312 sectors
```

Issue the three `iscsiadm` commands:

```
[opc@instance1 ~]$ sudo iscsiadm -m node -o new -T iqn.2015-
12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789 -p
169.254.2.2:3260
New iSCSI node [tcp:[hw=,ip=,net_if=,iscsi_if=default]
169.254.2.2,3260,-1 iqn.2015-12.com.oracleiaas:3a108842-7f63-431e-
82db-11adffca1789] added

[opc@instance1 ~]$ sudo iscsiadm -m node -o update -T iqn.2015-
12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789 -n node.startup
-v automatic

[opc@instance1 ~]$ sudo iscsiadm -m node -T iqn.2015-
12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789 -p
169.254.2.2:3260 -l
Logging in to [iface: default, target: iqn.2015-
12.com.oracleiaas:3a108842-7f63-431e-82db-11adffca1789, portal:
169.254.2.2,3260] (multiple)

Login to [iface: default, target: iqn.2015-12.com.oracleiaas:3a108842-
7f63-431e-82db-11adffca1789, portal: 169.254.2.2,3260] successful.
```

Now the `fdisk -l` command shows the newly attached `/dev/sdb` disk.

```
[opc@instance1 ~]$ sudo fdisk -l |grep /dev
WARNING: fdisk GPT support is currently new, and therefore in an
experimental phase. Use at your own discretion.
Disk /dev/sda: 50.0 GB, 50010783744 bytes, 97677312 sectors
Disk /dev/sdb: 53.7 GB, 53687091200 bytes, 104857600 sectors
```

You can now partition the disk, create a file system on the disk, and mount the disk. The following example creates an ext4 file system on the entire `/dev/sdb` device, creates a mount point, mounts the file system, and shows the mounted file systems:

```
[opc@instance1 ~]$ sudo mkfs -t ext4 /dev/sdb
mke2fs 1.42.9 (28-Dec-2013)
/dev/sdb is entire device, not just one partition!
Proceed anyway? (y,n) y
...
Writing superblocks and filesystem accounting information: done
[opc@instance1 ~]$ sudo mkdir /extra_disk
[opc@instance1 ~]$ sudo mount /dev/sdb /extra_disk
[opc@instance1 ~]$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
devtmpfs        14G     0    14G  0% /dev
tmpfs           14G     0    14G  0% /dev/shm
tmpfs           14G   17M   14G  1% /run
tmpfs           14G     0    14G  0% /sys/fs/cgroup
/dev/sda3       39G   1.8G   37G  5% /
/dev/sda1       512M  9.8M  502M  2% /boot/efi
tmpfs           2.8G     0    2.8G  0% /run/user/0
tmpfs           2.8G     0    2.8G  0% /run/user/1000
/dev/sdb        50G   53M   47G  1% /extra_disk
```

On Oracle Linux instances, it's important to include the `_netdev` and `nofail` options on every non-root block volume in `/etc/fstab` or the instance can fail to launch as the OS tries to mount the volumes before the iSCSI initiator has started.

Oracle Cloud Computing Resources

Refer to the following resources for further information about Oracle Cloud Computing:

- <http://www.oracle.com/cloud>
- <https://cloud.oracle.com/home>
- <https://cloud.oracle.com/cloud-infrastructure>
- <https://docs.us-phoenix-1.oraclecloud.com/Content/home.htm>
- <https://education.oracle.com/cloud>



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Quiz

Q

What of the following statements are true? (Select all that apply.)

- a. Oracle Cloud Infrastructure are physically hosted in Regions and Availability Domains.
- b. Oracle Cloud Infrastructure are physically hosted in Tenancies and Compartments.
- c. When you sign up for Oracle Cloud Infrastructure, Oracle creates a tenancy for your company.
- d. Compartments allow you to organize and control access to your cloud resources.



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Quiz



What of the following statements about Oracle Cloud Infrastructure instances are true?
(Select all that apply.)

- a. You need to set up at least one Virtual Cloud Network (VCN) before you can launch instances.
- b. Bare metal compute instances run on bare metal servers without a hypervisor.
- c. Managed Virtual Machine (VM) instances are also available for workloads that don't require dedicated physical servers or the high-performance of bare metal instances.
- d. When launching an instance, you also need to select an image, a shape, a private IP address, and public IP address.



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Summary

In this lesson, you should have learned how to:

- Describe Infrastructure as a Service (IaaS)
- Describe Oracle Private Cloud Appliance
- Describe Oracle OpenStack
- Describe Oracle Cloud Infrastructure
- Describe Key Concepts and Terms Used in Oracle Cloud Infrastructure
- Describe Oracle-Provided Images and Available Shapes in Oracle Cloud Infrastructure
- Describe the Task Flow to Launch an Oracle Cloud Infrastructure Instance
- Create an Oracle Cloud Infrastructure Virtual Cloud Network and Subnet
- Launch an Oracle Cloud Infrastructure Instance
- Attach an Oracle Cloud Infrastructure Block Storage Volume to an Instance



ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.