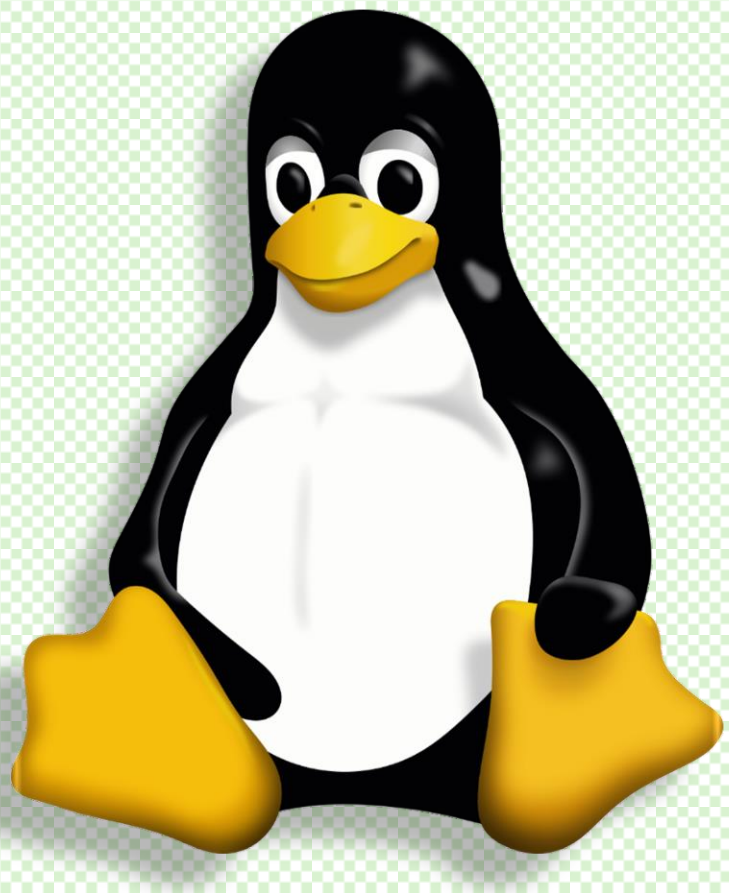
Essential Linux Commands for MLOps & DevOps



Day 2 of 9

Table of Contents

1. Introduction

Welcome to Day 2 of our Basic Linux Commands for MLOps and DevOps Engineers tutorial series.

Today, we'll dive into File Management—a crucial skill for efficiently handling data and configuration files on Linux systems.

Whether you're dealing with log files, data sets, or configuration files, mastering file management commands will streamline your workflow.

2. Creating and Deleting Files and Directories

Let's start with basic operations: creating and deleting files and directories.

Creating Files

The touch command is used to create a new empty file.

```
# Creating a new file
!touch example.txt
```

This command creates a file named example.txt in the current directory. You can verify its creation using the ls command:

```
# Listing files in the current directory

!ls
```

Creating Directories

The mkdir command is used to create a new directory.

```
# Creating a new directory
!mkdir my_directory
```

You can create nested directories (directories within directories) by using the -p option.

```
# Creating nested directories
!mkdir -p parent_directory/child_directory
```

Deleting Files & Directories

The rm command is used to delete files. Be careful with this command, as it does not move files to a trash bin—it permanently deletes them.

```
# Deleting a file
!rm example.txt
```

To delete a directory and its contents, use the -r (recursive) option.

```
# Deleting a directory and its contents
!rm -r my_directory
```

3. Copying and Moving Files and Directories

Copying Files

The cp command is used to copy files. To copy a file from one location to another, you can use:

```
# Creating a new file named source_file.txt
!echo "This is a sample file." > source_file.txt

# Copying a file
!cp source_file.txt destination_file.txt
```

To copy a directory and its contents, use the -r option.

```
# Copying a directory
!cp -r source_directory/ destination_directory/
```

Moving Files

The mv command is used to move or rename files and directories.

```
# Creating a new directory as the destination
!mkdir -p /content/destination_directory

# Creating a new file named source_file.txt
!echo "This is a sample file." >
/content/source_file.txt

# Verifying the destination directory was created
!ls /content/

# Moving the file to the destination directory
!mv source_file.txt /content/destination_directory/

# Verifying the file was moved
!ls /content/destination_directory/
```

This command moves source_file.txt to the specified directory. You can also use mv to rename files or directories:

```
# Creating a new file named source_file.txt
!echo "This is a sample file." > /content/old_name.txt

# Renaming a file
!mv old_name.txt new_name.txt


# Verifying the destination directory was created
!ls /content/
```

4. Renaming Files and Directories

Renaming files and directories is done using the mv command, as shown above.

```
# Creating a new directory as the destination
!mkdir -p /content/old_directory_name/

# Verifying the destination directory was created
!ls /content/

# Renaming a directory
!mv old_directory_name/ new_directory_name/

# Verifying the destination directory was created
!ls /content/
```

This changes the name of old_directory_name to new_directory_name.

5. Finding Files and Directories

The find command is a powerful tool to search for files and directories. You can search by name, type, size, and other attributes.

Finding Files by Name

```
# Finding a file by name
!find /content/ -name "new_name.txt"
```

This command searches for a file named filename.txt starting from /path/to/search/.

Finding Directories

0s

```
# Finding a directory by name
!find /content/ -type d -name "new_directory_name"
```

6. Viewing File Contents

Viewing the contents of a file is often necessary to check configurations, logs, or data.

'cat' Command

The 'cat' command displays the entire content of a file.

0s

```
# Viewing file content
!cat example.txt
```

'less' Command

The 'less' command allows you to view the content of a file page by page.

```
# Viewing file content page by page
!less example.txt
```

'head' and 'tail' Command

These commands are useful to view the beginning or the end of a file.

```
# Viewing the first 10 lines of a file
!head example.txt

# Viewing the last 10 lines of a file
!tail example.txt
```

You can also specify the number of lines to display:

```
# Viewing the first 5 lines
!head -n 5 example.txt

# Viewing the last 5 lines
!tail -n 5 example.txt
```

7. Conclusion

Today, we covered the essential file management commands in Linux, which are vital for MLOps and DevOps engineers.

Mastering these commands will allow you to efficiently handle files and directories, an essential skill in managing large datasets and configuration files.

In the next tutorial, we'll dive into file management, including how to create, move, copy, and delete files and directories. Stay tuned!

8. Practice Tasks

- Create a directory structure for a project and populate it with files.
- Copy and move files between directories.
- Find specific files in your directory structure.
- View the contents of various files to understand their structure.
- By practicing these tasks, you will gain hands-on experience in managing files on a Linux system.

Stay tuned for Day 3, where we will explore how to view and edit files using command-line tools.

Example Google Colab Notebook