**Essential Linux Commands for Data Engineers: A Comprehensive Guide**

As a Data Engineer, proficiency in Linux commands is crucial for managing data pipelines, handling files, and working with distributed systems. This guide covers essential Linux commands with practical examples relevant to data engineering tasks.

**Navigation Commands**

**Working Directory Management**

# Show current directory

pwd

# Output: /home/dataeng

# Go to home directory

cd ~

# Go up one level

cd ..

# Go to specific directory (absolute path)

cd /data/warehouse

# Go to relative path

cd ./raw\_data

**Directory Listing**

# List files

ls

# Detailed listing with permissions and timestamps

ls -l

# Sort by time (newest first)

ls -lt

# Sort by time (oldest first)

ls -ltr

# Show hidden files

ls -a

# Recursive listing (useful for data directories)

ls -R

**File Operations**

**Creating and Copying Files**

# Create empty file

touch data.csv

# Copy file

cp source.csv destination.csv

# Copy with preservation of metadata (important for data lineage)

cp -p source.csv destination.csv

# Copy directory recursively

cp -R /source\_dir /target\_dir

**Moving and Renaming Files**

# Move file

mv source.csv /new/location/

# Rename file

mv oldname.csv newname.csv

# Move multiple files to directory

mv file1.csv file2.csv target\_dir/

**File Removal**

# Remove file

rm filename.csv

# Remove directory and contents

rm -r directory\_name

# Force remove (use with caution!)

rm -f locked\_file.csv

**File Viewing and Editing**

**Viewing File Contents**

# View entire file

cat data.csv

# View first 10 lines

head data.csv

# View first n lines

head -n 20 data.csv

# View last 10 lines

tail data.csv

# View last n lines

tail -n 20 data.csv

# Follow log file in real-time (crucial for monitoring data pipelines)

tail -f pipeline.log

**File Editing**

# Open file in vi editor

vi data.csv

# Basic vi commands:

# i - enter insert mode

# esc - exit insert mode

# :w - save

# :q - quit

# :wq - save and quit

# :q! - quit without saving

**File Permissions**

**Understanding Permissions**

# View file permissions

ls -l

# Output: -rw-r--r-- 1 dataeng datagrp 1024 Jan 4 10:00 data.csv

# Permission structure:

# r (read) = 4

# w (write) = 2

# x (execute) = 1

# Change permissions

chmod 644 data.csv # Owner: rw-, Group: r--, Others: r--

chmod 755 script.sh # Owner: rwx, Group: r-x, Others: r-x

**Search and Pattern Matching**

**Using grep**

# Search for pattern in file

grep "ERROR" pipeline.log

# Case insensitive search

grep -i "error" pipeline.log

# Recursive search in directory

grep -r "FAILED" /logs/

# Count occurrences

grep -c "SUCCESS" pipeline.log

# Find files by name find /data -name "\*.csv" # Find files modified in last 24 hours find /data -mtime -1 # Find and execute command find /data -name "\*.tmp" -exec rm {} \;

**Data Processing Commands**

**Basic Data Processing**

# Count lines in file

wc -l data.csv

# Sort data

sort data.csv > sorted\_data.csv

# Remove duplicates

sort data.csv | uniq > unique\_data.csv

# Split large files

split -l 1000000 large\_file.csv chunk\_

**Data Transformation**

# Extract specific columns (using cut)

cut -d',' -f1,2 data.csv > subset.csv

# Replace text

sed 's/old/new/g' data.csv > modified.csv

# Filter rows

awk -F',' '$3 > 1000' data.csv > filtered.csv

**Common Data Engineering Scenarios**

**Log Analysis**

# Find error patterns in logs

grep "ERROR" app.log | cut -d' ' -f1,2 | sort | uniq -c

# Monitor failed jobs

tail -f pipeline.log | grep --line-buffered "FAILED"

# Calculate success rate

echo "Success rate: $(grep -c "SUCCESS" job.log)/$(wc -l < job.log)"

**Data Pipeline Operations**

# Monitor disk usage

du -h /data/warehouse

# Check file counts

ls -1 /data/input | wc -l

# Verify file integrity

md5sum data.csv > checksum.txt

**Compression and Archiving**

# Compress files

gzip large\_file.csv

# Create tar archive

tar -czf archive.tar.gz /data/files/

# Extract tar archive

tar -xzf archive.tar.gz

**Best Practices for Data Engineers**

1. **Always Use Absolute Paths in Scripts**:

/data/warehouse/raw/input.csv # Better than relative paths

1. **Check Command Success**:

command && echo "Success" || echo "Failed"

1. **Use Variables for Repeated Values**:

DATA\_DIR="/data/warehouse"

cd "$DATA\_DIR"

1. **Create Backup Before Operations**:

cp data.csv data.csv.bak

1. **Monitor Resource Usage**:

df -h # Check disk space

top # Monitor processes

**Tips for Remote Operations**

# Secure copy files between servers

scp data.csv user@remote:/data/

# Execute remote commands

ssh user@remote "ls -l /data/"

# Tunnel for database connections

ssh -L 3306:localhost:3306 user@remote