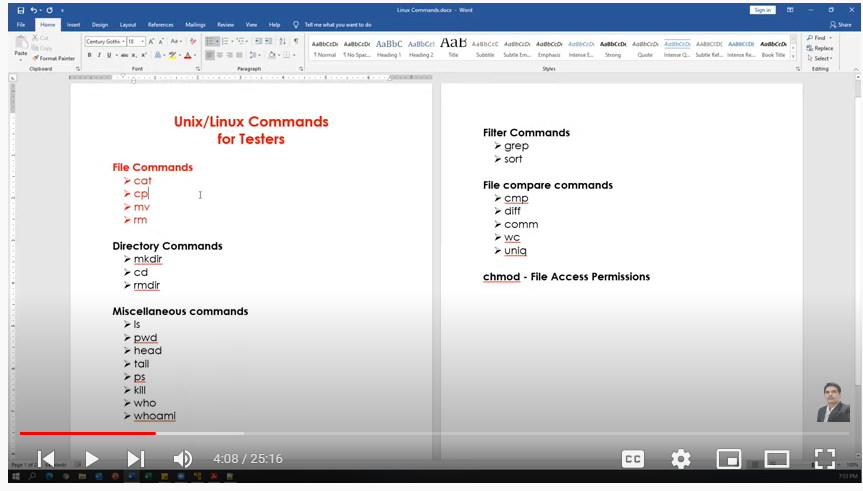
**Unix/Linux for Testers**

****

**Miscellaneous commands:**

(i) "Clear" is the command to clear the screen in Linux not *cls.*

**File Commands:**

**cat**

Creating the new file

Display content of the file

Concatenating more than one file

Appending data to the existing file

cp - Copy contents from a file to another file(source to destination)

mv

Renaming a file(changing name of the file)

Renaming a directory

Moves files from one directory to another directory

rm - Delete/remove a file or directory (only if directory is empty).

Note: (i) we can also create file using "touch" command. Difference between "cat" vs "touch".

(ii) File can be edited using "Vim" editor.

cat

cat >testing.txt # to create file

manual testing

automation testing

(ctrl+d) for exit

cat testing.txt # for display

cat yourdirecory/automation.txt # to display anything inside directory

cat testing.txt testing123.txt # for concatenate

cat >>testing.txt # for amend

cp

cp testing.txt testing\_new.txt

# Copy contents from a file to another file(source to destination)

mv

mv testing.txt automation.txt # re-name a file

mv mydir yourdir # re-name a directory

mv automation.txt yourdir # moving a directory

rm

rm testing.txt # to delete a file

rm "directory name" # to delete a directory when it's empty

rm -r "directory name" # to delete a directory forcefully when some files in it

" -r" is used for recursive.

Note: we can also use "rmdir" command to delete a directory if it is empty

**Directory Commands**

mkdir - create directories and subdirectories.

mkdir testdir

create multiple directories at one time:

mkdir testdir1 testdir2 testdir3

create several subdirectories at one time:

mkdir -p world/countries/states

-------------------------------------------------------------------

cd - changing/closing directory

cd .. : go back to one level

cd /testdir1/testdir2/testdir3 → changing directory

pwd : it prints present directory

cd ~ Move to users home directory from anywhere.

------------------------------------------------------------------

rmdir - Remove the directory if it is empty (works only if directory empty)

rmdir world // error (because "world" directory is not empty)

\*\* In this case we can use **"rm -r"** command

rm -r world

rmdir testdir1 testdir2 testdir3

**touch, pwd & ls Commands**

touch - to create a file

touch myfile.doc

touch /myhiddenfile.txt #to create a hidden file

---------------------------------------------------------------------------------------------------------------------------

ls : List Files

ls -l : (long list) shows file or directory, size, modified date and time, file or folder name and owner of file and its permission.

ls -a : view hidden files

touch .myfile.txt

ls -a

ls -l -a : detailed listing files along with hiddwn files

ls -F : will add the ‘/’ Character at the end each directory.

ls -r : display files and directories in reverse order.

ls -R : displays directories along with sub subdirectories

ls -lS : ("S" in upper case) displays file size in order, will display big in size first.

ls -l Documents : list files under directory Documents.

(I need to check this command in detail in future)

wild card characters

------------------------------

? Single character

\* Multiple characters

[ ] Range of values

ls ?.\*

Output: a.doc b.doc c.doc x.txt y.txt

ls ?.doc

Output: a.doc b.doc c.doc

ls ?.txt

Output: x.txt y.txt

ls a\* Displays files which are starting with 'a'

Output: abc.doc a.doc

Range(Displays files starting with a to z)

--------------------------------------------------------

ls [a-z]\*.\*

Output:



ls [a-c]\*.\*

Output:



ls [a-z]\*.txt

**head, tail, more & less Commands**

**Note:** Instead of "cat" command we have these following commands for display.

**head :** to display specified number of lines from top of the file.

--------

**head cities.txt**

\* Display 10 lines from top of the file.

\* 10 is the default value for head command

**head -n 15 cities.txt** (or) **head -15 cities.txt** # will show first 15 lines of the text

**head -n 5 cities.txt** # will show first 5 lines of the text

**tail** : to display specified number of lines from bottom of the file.

----------------

**tail cities.txt**

\* Display last 10 lines from the file.

\* 10 is the default value for tail command

**tail -n 15 cities.txt** (or) **tail -15 cities.txt** # will show last 15 lines of the text

**tail -n 5 cities.txt** # will show last 5 lines of the text

Display the lines from 10 to 15

**head -15 cities.txt | tail -6**

Display the lines from 20 to 30

**head -30 cities.txt | tail -11**

**ls - l** Display List of files and directories

**ls -l | head -5** Display Top 5 files and directories

**ls -l | tail -5** Display Top 5 files and directories

**more :** Display content page by page. **Forward directional**. Can't go to the previous page.

(Next page: press "space bar", Next line: press "Enter" button, Exit: press "q" button)

**more cities.txt**

**less** : Display content page by page in **both directions** means *next page or to previous page.*

(Next page: press "space bar", Next line: press "Enter" button, Exit: press "q" button)

(Previous Page: Press upper key "↑")

**less cities.txt**

-----------------------------------------------------------

**ls -l | more**

**ls -l | less**

**who, whoami, hostname, uptime, cal & date Commands**

who - Display how many users have connected to the Linux system

whoami - Displays the username of the current user.

hostname

----------

**hostname** : Print the host name of the system

**hostname -i** : print IP address of the computer

uptime

-----------

It is used to find out how long the system is active (running).

the current time,

the amount of time system is in running state

number of users currently logged into

the load time for the past 1, 5 and 15 minutes respectively.

---------------------------

cal : Display current month's calendar

cal 2021 : display calendar of specified year

cal 3 2021 : display specific month calendar in a year ( 3 rd month in 2021)

cal -3 : display previous, current and next month

cal -y : Display calendar of current year

cal -m10 : Display calendar of month n the current year

----------------

date

----------

date

date "+%Y" : Display year

date "+%m" : Display month(Number)

date "+%d" : Display date (Number)

date "+%d-%m-%Y" output: 05-12-2020

date "+%d/%m/%Y" output : 05/12/2020

date "+%d/%m/%y" Output: 05/12/20

date "+%a" display short weekday name (e.g., Mon)

date "+%A" display Full weekday name (e.g., Monday)

date "+%b" display short month name (e.g., Jan)

date "+%B" display Full month name (e.g., January)

date "+%H" Current hour in 24-format

date "+%I" Current hour in 12-format

date "+%M" Minutes

date "+%S" Seconds

date "+%H:%M:%S" Print Current time in HH:MM:SS Format [in 24-Hrs format]

date "+%I:%M:%S" Print Current time in HH:MM:SS Format [in 12-Hrs format]

date "+Today's Date & time is: %d-%m-%Y %I:%M:%S" [in 12-Hrs Format

Output: Today's Date & time is: 05-12-2020 11:46:55

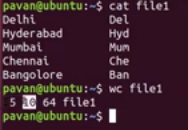
date "+Today's Date & time is: %d-%m-%Y %I:%M:%S" [in 24-Hrs Format]

more formats of dates...

date --help

**wc, sort & uniq Commands**

wc --- to count total numbers of lines, word and characters

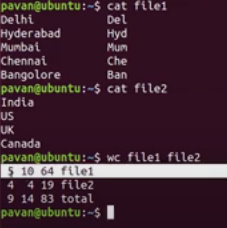


5 ---> Numbers of line

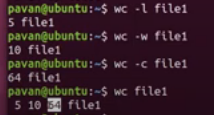
10 ---> Numbers of words

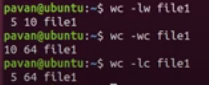
64 ---> Numbers of characters

"wc" command can be used for multiple files



"wc" command can be used with "l", "w" and "c". "l" for no. of lines, "w" for no. of word and "c" for no. of characters.





sort --- sort the data but only for Display Purpose. It doesn't change the original data.

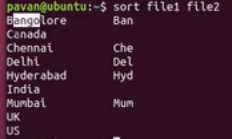


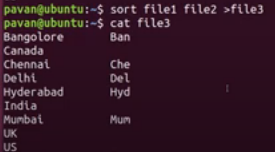
"Sort" command also sort the numbers in the same way.

After sorting the data we want to store the data in some new file. Please use "re-directional" arrow or "right arrow"



Sort the data from multiple files.





Sort the data in reverse order: *sort -r numbers.txt*

uniq command is used to remove duplicate values and stores only unique value.

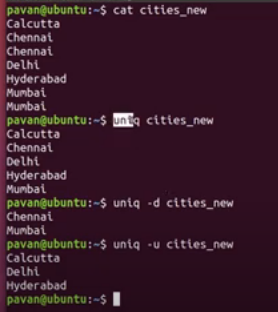
to use "uniq" command data should be in sorted order.

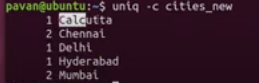
uniq cities\_new

uniq -d cities\_new #d ---> for duplicate values

uniq -u cities\_new #u ---> for unique values

uniq -c cities\_new #c ---> how many times any word repeated





# cmp, diff & comm Commands

gedit

gedit file 1 (If you save it here the file will be saved)

clear

1. Cmp

Text

Description automatically generated

“Cmp” command compared byte by Byte. Each character here is 2 byte.

1. Diff

Text

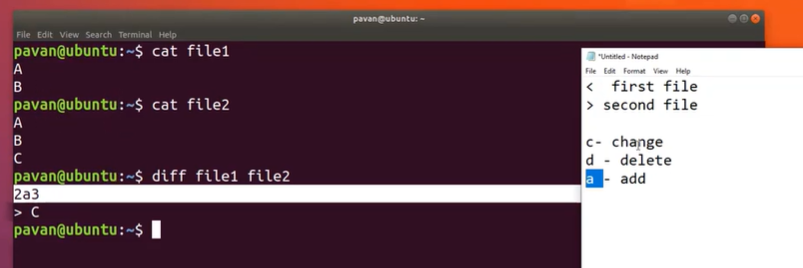
Description automatically generated

Graphical user interface, text

Description automatically generated

Text

Description automatically generated



1. Comm

Text, letter

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

# Part 8 - Unix/Linux for Testers | Input, Output & Error Re-Direction

Diagram

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, email

Description automatically generated

Text

Description automatically generated

Double Greater than (>>) is used for appending data.

Text

Description automatically generated

For Output Re-direction “1” is optional.

Text

Description automatically generated

A picture containing text

Description automatically generated

Calendar

Description automatically generated with medium confidence

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Here “2” is mandatory

Error Direction:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated

Hero “zero (0)” is optional.

Text

Description automatically generated with low confidence

Text

Description automatically generated

Text, letter

Description automatically generated

In Linux everything is File. Even Terminal is also File. Type “tty” to display path of the Terminal.

Graphical user interface, application

Description automatically generated

# Part 9 - Unix/Linux for Testers | Piping and Usage of tee and xargs

Piping:

We can pass output of one command as input for next command is called Piping.

Using piping multiple commands works together.

We can use Piping using Vertical Bar (I)

Command1 -🡪Output -🡪 Command2 -🡪 Final Output

Ex1:

ls -l /etc | wc -l

Ex2:

ls – l /etc | more

Ex3:

ls -l /etc | head -s

Ex4:

ls -l /etc | wc

Ex5:

The output of ls command should be saved to output.txt file and should be provided as input to wc command.

ls -l /etc > output.txt | wc

* In the middle of piping if we use redirection then it breaks piping.

tee:

ls -l /etc > tee output.txt | wc

Ex6:

The output of ls command should be saved to output.txt file and should be displayed.

ls -l /etc > tee output.txt

Ex7: tests.txt contains filenames. Read file names from tests.txt and remove each file.

cat tests.txt ==🡺 returns standard output test1, test2, test3

rm test1, test2, test3 ==🡺 remove 3 files

command1 🡪 command2 --🡪 Final Output

cat tests.txt | xargs rm

Ex8:

Display the output of date command by using echo command with piping concept.

date

echo “welcome”

date | xargs echo

# Part 10 - Unix/Linux for Testers | Regular Expressions

* Regular expression is also called regex
* Regular expression is a pattern for matching string
* Regex can be used with different command like ls, rm, grep, sed, rename and many more

1. To list out all the files present in the current directory.

ls \*

1. To list out all files with some extension

ls \*.\*

1. List out all .txt files

ls \*.txt

1. List out only .java files

ls \*.java

1. List out all the files starts with “t”

ls t\*

1. List out all files starts with “c” and ends with “s”

ls c\*s

1. List out all files where filename contains only 2 characters

ls ??

8.list out all files where file name contains only 2 characters, and first character should be “x”

ls x?

1. List out all files where file name contains only 5 characters

ls ?????

1. List out all files where filename contains at least 3 characters

ls ???\*

1. List out all the files where file name starts with s or t or r

ls [str]

1. List out all the files where file name should not start with “s”, or “t” or “r”

ls [!str]

1. List out all the files where file name starts with lower case alphabet (a,b,c, …..z)

ls [a-z]\* -----🡪 not works in Ubuntu

ls [[:lower]]\*

1. List out all the files where file name starts with upper case alphabet (A,B,C, ….Z)

ls [A-Z]\* 🡪 not works in Ubuntu

ls [[:Upper]]\*

15.List out all the files where file name starts with digit (0,1,2,3 ….9)

ls [0-9]\* --🡪 works in all the Unix/ Linux flavors.

ls [[:digit]]\*

1. List out all the files where

First letter should be Upper case

Second letter should be digit and

Third Letter should be lower case letter

ls [A-Z][0-9][a-z]\*

ls [[:upper:]][0-9][[:lower:]]

1. List out all files starts with special symbol

ls [![:alnum:]]\*

1. List out all the files with .java or .py extension

ls {\*.java, \*.py}

1. Copy all files starts with digit to dir1 directory

cp [0-9] \* dir1

1. Move all files starts with alphabet and .txt extension to dir2 directory

mv [[:alpha:]]\*.txt

# Part 11 - Unix/Linux for Testers | grep | egrep | fgrep commands

Grep (Global Regular Expression Pattern)

This command is used for searching a required pattern in file.

Ex1:

------

grep “Chennai” cities

grep “Delhi” cities

Ex2: grep command option:

-------------------------------------

-i Ignores case sensitiveness in searching option

-n displays line numbers along with matched pattern

-c counts number of times a searching pattern exists

-v (verbose) Displays lines that does not match with the pattern

-l Displays the files names that matches the pattern (we can just display the files that contains the given string/pattern)

grep with regular expression

----------------------------------------

Ex1: Display all lines that search with “D”

^ Represent starting of the line

grep ‘^D’ cities

Ex2: Display all lines that ends with ‘i’

$ Represent end of the line

grep ‘i$’ cities

Ex3: Display all lines that ends with “bad”

grep ‘bad$’ cities

Ex4: Display all lines that contain any of the letter’s A, B, C, D

grep ‘[A-D]’ cities

grep ‘[ABCD]’ cities

Ex5: search for vowels in the file a,e,i,o,u

grep ‘[aeiou]’ cities

Ex6: search for consonants in the file (other than a,e,i,o,u)

Grep ‘[^aeiou]’ cities

Ex7: Search for multiple patterns

-e Search for multiple patterns

grep -e “Delhi” -e “Chennai”

Instead of “e” we can use egrep command

egrep “(Delhi|Chennai)” cities

Difference between grep and egrep:

------------------------------------------------------

egrep “(Delhi|Chennai)” cities

grep “(Delhi|Chennai)” cities

grep command understands patterns but not all

egrep command understand all the patterns

-F search for fixed patterns (No Pattern)

grep -F “Delhi

>Mumbai

>Hyderabad” cities

Instead of F we can use fgrep command. (Fixed string Global Regular Expression pattern)

fgrep “Delhi

* Hyderabad
* Mumbai” cities

fgrep command can’t understand the patterns. Understand fixed strings.

grep with piping

---------------------------

Command1 | Command 2 --🡪 result

Ex1:

-----

cat cities

ls -l | grep ‘cities’ | wc -l

ls cities | xargs wc

# Part 12 - Unix/Linux for Testers | chmod command | File Access Permissions

chmod command

-----------------------------

3 types of roles

owners/users (u)

groups (g)

others (o)

3 types of permission

read – (r}

write – (w)

execute – (e)

rw- r—r--

rw- First 3 represents user/owner permission

r—Next 3 represents group permissions

r—Last 3 represents other permissions

Project: XYZ

Module1 Module2 Module3

A,B,C D,E,F G,H,I,J

A is created a file sample.txt

A ----- user/owner (person created file)

B,C ----- group (persons working in the same module)

D,E,F,G,H,I,J ----Others (persons working in the same module)

1. Symbolic/Text Method
2. Numeric method

Symbolic/Text Method

---------------------------------

Ex1: write a command to add execute permission to owner of the file

chmod u+x sample.txt

Ex2: write a command add execute permission to owner and add read, write permissions to group and others

u –x

o – r w

chmod u+x, g+rw, o+rw

chmod u+x, go+rw sample.txt

Ex3: write a command remove read permissions from group and others

chmod g-r,o-r sample.txt

chmod go-r sample.txt

Ex-4: chmod u-w, g-w, o-r sample.txt

Write permission cancelled from owner

Write permission cancelled from groups

Write permission cancelled from others

Ex-5: chmod u+rwx, g+rwx, o+r sample.txt

Numeric Method:

-----------------------------

3 types of permissions

Read (r) – 4

Write (w) – 2

Execute (e) - 1

---------------------------------------------

7

-------------------------

Ex1 : chmod 000 sample.txt

No Permission to owner, group and others

Ex2: chmod 777 sample.txt

All permission given to owner, group and others

Ex3: chmod 444 sample.txt

Read permission given to owner, group and others

Ex4: chmod 600 sample.txt

Read (4), Write (2), permission given to owners

No Permission to groups and others

Ex5: chmod 664 sample.txt

Read, Write permissions given to owners, groups, and Read permission given to others

Ex6: chmod 111 sample.txt