Linux Commands

File Commands

cat
cp
mv
rm

Director Commands

mkdir
cd
rmdir

Filter Commands

grep
sort

miscelleneous commands

ls
pwd
head
tail
PS
kill
who
whoami

file Compare commands

cmp
diff
comm
wc
uniq

Chmod - File Access Permissions

File Commands

① cat ⇒ ① can create new file
② can show content from file
③ can concatenate multiple files
④ appending data to the existing file

ls ⇒ listing command (list all directories)

clear ⇒ clear the screen

```
cat > test.txt # creating file
```
manual test
Automan test } ⇒ enter data in file

cntr+d ⇒ come out from cmd prompt

```
cat test.txt # display content of file
```

```
cat > test1.txt # create another file
```
API
Jmeter
cntr+d

```
cat test.txt test1.txt # display content from both file
                       (concatenate)
```

```
Cat >> test.txt # append data in file
```
git
Jenkins

② cp → copy command
copy content from one file to another file.

```
CP test.txt test123.txt # copy content from test to test123
   ↑ source file  ↑ target file
```

③ mv → ① To rename existing file
② To rename directory (contains multiple files) (folders)
③ move file from one directory to another directory.

① content will same just change file name.

```
mv test.txt manual.txt # test.txt file will rename with manual.txt
```
ls

```
mkdir   mydir   # create directory
```

```
mv mydir  dir  # Rename directory of mydir
                 with dir
```

```
PWd ⇒ present working directory
```

```
mv  manual.txt  dir # move manual file to
                      another directory
```

```
cd  dir  # to change directory
```

```
cd..  # change directory
        it will back to one level
```

④ rm ⇒ ① remove / delete file or directory

② can directly delete file but for directory it should empty then also we can delete directory.

```
rm  test.txt  # remove file
```

```
rm  mydir  # remove directory which is having
             some files
  # cannot remove
```

```
rm -r  mydir  # can remove dir if it
                contains file
```

Directory Commands

① mkdir ⇒ ① make directory or sub directory

```
mkdir testdir # create directory
```

```
mkdir test1 test2 test3 # create multiple directory in current dir
```

```
mkdir -p world/countries/states # create multiple sub directory
```

(world: parent; countries: sub directory)

How to check ⇒ cd world
ls
cd countries
ls

② cd ⇒ change directory & can navigate to diff sub directory & can also go to the root directory

↳ cd countries
↳ cd states
↳ pwd
↳ cd ~ # move to root/home directory
↳ pwd
↳ cd .. # go back to one level

cd world
ls → whatever files available in world will display

③ rmdir ⇒ ① remove directory if directory is empty

```
rmdir testdir # remove directory
rmdir world # will show error because having files
rm -r world # remove even folder contains subfiles
rmdir test1 test2 test3 # remove all directory at once should not have subdirectories or files
```

④ touch → To create empty file
→ create hidden file

```
touch myfile.doc #create empty file
touch .myhiddenfile.txt # create hidden file
ls -a # display all files/directory along with hidden files
```

⑤ ls → ① give all files [list the files]

```
ls -l # to see details of each files
ls -a # all files along with hidden files
ls -l -a : # details info of files includes hidden files.
ls -F # can recognize files
      ↑ add slash end of each directory
```

`ls -r` # all files and directories displayed in reverse order

`ls -R` # displayed directories with subdirectory (details info)

`ls -ls` # displayed each and every size of file (highest size will be displayed first)

`ls -l dirname` # displayes all files which is available in dirname

`ls filename` # display files which is available
`ls foldername` in filename/foldername

Wild card characters

? → single character
* → multiple character
[ ] → Range of character

`ls ?.*` # filename should be single character (All the filename is having single character) but can have different extensions

(? : single character, . : extension, * : multiple character)

`ls *.doc` # files contains multiple characters (can be anything)

`ls ?.txt` # file name is single character with txt extensions

`ls 'a*.txt` # a is single character after that any characters with txt extensions.

`ls [a-z]*.*` # list out files whose names comes under a-z (file can be any no. of characters) with any extensions.

ls [a-c]*.* # only a-c files name will contain with multiple characters

cat ⇒ drawback → if content is more in file then its not possible to see. (can see but need to navigate top to bottom)

Head ⇒ ① use to display specific no. of line from the top of the file. default value = 10 lines

```
head city.txt # bydefault 10 lines from top of file displayed
head -n 5 city.txt # 5 lines display from top of file
head -n 20 city.txt # display top 20 lines
head -20 city.txt # same as above
```

Tail ⇒ ① It will display last 10 lines from files (bottom 10 lines)

```
tail city.txt # display last 10 lines from file
tail -n 20 city.txt # display last 20 lines from file
tail -20 city.txt # same as above
```

Head tail ⇒ can get both files

```
head -15 city.txt | tail -6 city.txt
tail -6 city.txt
```

pipe symbol

it will apply on previous getting

```
# print 20-30 from 1-10 files
head -30 test.txt | tail -10 test.txt
```

Head tail with ls (can use for files or folders).

ls -l | head # display top 10 files/directory from given list

ls -l | head -3

(whatever output is getting from ls cmd passing to)

ls -l | tail -10 # display bottom 10 files from list

ls -l | tail # same as above

ls -l | tail -5

more → ① To display content of file page by page then use more command. (can go Next page)
② can't navigate to previous page.

more city.txt # display 1st page

press space # display 2nd page

press space # display 3rd page

q # get out from and prompt

Enter # it will go line by line

(type from keyboard)

less ⇒ ① display content page wise
② difference is can navigate both page (pre & next)

less city.txt

press upper arrow key # previous page

Who ⇒ ① show how many users are connected to the linux
② linux/unix use ~~lot of~~ server mlc, lot of client mlc connected to server
③ multiple user connected

whoami ⇒ current user name of current logged in user

hostname ⇒ hostname of computer
If we know hostname or IP address, we can connect with server mlc.

hostname -i # display IP address of current mlc
127.0.1.1 # default IP (No internet connect?)

uptime ⇒ ① How long ~~use~~ system is active or how long system is running. (if u want know)

cal ⇒ by default show you current month calender

cal 2022 # display 2022 calender

cal 3 2022 # 3rd month of 2022 will display

can see previous or next year calender

cal -3 # display pre, current, next month of current year

cal -y # calender of current year

cal -m10 # calender of specific month/year increment

* Date ⇒

date # current week name, month, time

date "+%y" # current year

date "+%m" # current month

date "+%d" # current date

date "+%d-%m-%Y" # display date-month-year

date "+%d/%m/%Y" # date/month/year

date "+%y" # last 2 digit of year

date "+%a" # current day

date "+%A" # display full name of day

date "+%b" # current month (In short form)

date "+%B" # current month full name

* Time

date "+%H" # current hour (24 hr format)

date "+%I" # current hour (12 hr format)

date "+%M" # current minutes

date "+%S" # current seconds

date "+%H:%M:%S" # current hr, min, sec

date "+ Todays date & time is %d:%m:%Y %H:%M:%S"

date --help # whatever format supported display

① WC ⇒ count total no. of lines, words, or, characters files

↳ ls

↳ cat file1

↳ wc file1

5 10 64 file1 # gives all info in given file

(5 ↑ No. of lines; 10 ↑ no. of words; 64 ↑ no. of characters)

↳ WC file1 file2 # display data from 2 diff files

5	10	64	file1
4	4	19	file2
9	14	83	total

↳ wc -l fil1 # display only no. of lines along with filename

↳ wc -w file1 # display only no. of words along with filename

↳ wc -c file # display no. of characters from given file

↳ wc -lw file # display lines & words

↳ wc -wc file # words characters

↳ wc -lc file # lines & characters

sort ⇒ to sort content of file

↳ ls

↳ cat file1

↳ sort file1 # display sort data in file

↳ cat file1 # data will be in same order (original data will not be affected)

↳ cat numbers.txt

↳ sort numbers.txt

↳ sort numbers.txt > number_new.txt # sorted data stored in new file

redirect previous data to new file

↳ cat number_new.txt #verify data

↳ sort file1 file2 # display sort data from both files

↳ sort file1 file2 > file3 # save sorted data of file1 file2 in file3

↳ cat file3 # display

by default sorted in Ascending order

Sort -r number.txt # sort in desc. order

Sort -r number.txt > file2.txt # store result of first file in 2nd file

uniq → only for display purpose can't affect original file

→ extract unique or duplicate data from file

② data should be in sorted order

↳ sort file1 > file2.

↳ cat file2

↳ uniq file2 # removed all duplicates & display

↳ uniq -d file2 # display duplicate data from file

unique -u file2 # display unique file
(duplicated data will not be displayed)

No. of ~~lines~~ times repeated data, display those data

↳ uniq -c file2

gedit

gedit #open UI

gedit file1 #open UI type data & save

gedit file2 # —11—

cat file1

cat file2

cmp ⇒ compare two files [byte by byte of the text information]

⇒ content converted into byte

⇒ compare text from file byte by byte

cmp file1 file2.

file1 file2 differ: byte 5, line 3

each characters is 2byte.

diff ⇒ → comparing 2 text files

→ display mismatches in file

gedit file1 → ABP

gedit file2 → ABMN

diff file1 file2

3, 4c3, 4

first file / second file

<C

<D

...

>M

>N

< - first file

> - second file

c - change

d - delete

a - add

gedit file1
a
b

gedit file2
A
B
C

cat file1
cat file2
diff file1 file2

2nd line from 1st line ← # 2a 3 → add third line
> c → add c in file 2

comm ⇒ it will give you more informatn
⇒ prerequisite → sort file first.
⇒ used to compare 2 sorted files
⇒ It provides o/p in 3 colm
⇒ In first colm displays unique lines of first file.
⇒ In second colm displays unique lines of second file.
⇒ In third colm displays common lines in 2 files.

sort file1 > file1_sorted
sort file2 > file2_sorted
cat file1_sorted # david John scott Tye
cat file2_sorted # Canedy John mary scott

comm file1_sorted file2_sorted

david
Canedy
John
Tye
mary
scott

chmod → ① provide permission to other user

Roles & permission ⇒ 3 types of roles in linux

① owners / users (u)
② groups (g)
③ others (o)

3 types of permissions

read - (r)
write - (w)
execute - (x)

ls -l # display all files & directories

rw- first 3 represents user/owner permission

r-- Next 3 represents group permission

r-- last 3 represents other permission

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 → other (persons working in other modules)

2 methods use in CHMOD command

① Symbolic / Text method

② Numeric method

① **Symbolic / Text method →**

write a cmd to add execute permission to owner of file

chmod u+x sample.txt

write a command add execute per to owner & add read, write permissions to group & others

u — x
o — rw

chmod u+x, o+rw, g+rw sample.txt

or

chmod u+x, go+rw sample.txt

-rwxrw-rw-

Remove read permission from group & others

chmod g-r, o-r file.txt

chmod go-r file.txt

chmod u-w, g-w, o-r sample.txt

u-w: given write perm to owner
g-w: given write per to owner group
o-r: given read perm to other

chmod u+rwx, g+rwx, o+rwx sample.txt

② Numeric method

3 types of permission

read - (r)	4
Write - (w)	2
execute - (x)	1
	7

chmod 000 sample.txt # remove all the permision from the file

owner group others
chmod 777 Sample.txt # all the permission to owner, group, other

ls -l sample.txt ⇒ to see

chmod 444 sample.txt # read permin to all user

chmod 600 Sample.txt # read write per to owner

600 → Combinat^n of 4 + 2 → read, write

chmod 664 Sample.txt # RW per to owner
RW per to group
read per to execute others

chmod 111 Sample.txt # e per to owner
—u— group
—u— other

Zip files / Extract files

tar ⇒ we can zip / extract files

Compress / zipping files ..

can do 3 diff formats ⇒ ① tar format
② gz → gun zip
③ Bz → bun zip

mydir → mydir.tar # zip file
mydir.tar.gz # (gun zip)
mydir.tar.bz2 # (bun zip)

tar -cvf

c → create a new .tar archive file
v → verbosely show the .tar file progress.
f → file name type of the archive file.

tar -cvf mydir.tar mydir # compress file in diff format

mydir.tar ↓ generated in zipped format

mydir ↑ this file will zipped (directory file)

mydir.tar ↑ which directory after compressing this dir it will generate this file

(extract करने पर zip file में)

tar -cvf mydir.gz mydir

tar -cvf mydir.bz2 mydir

delete mydir

↳ rm -r mydir

↳ ls

↳ extract files in zip formats

Extract / unzipping files

mydir.tar ⟶ mydir
mydir.tar.gz ⟶ mydir
mydir.tar.bz2 ⟶ mydir

tar -xvf
- x: extract
- v: verbos
- f: filename

extract directory from perticular files

tar -xvf mydir.tar target location
tar -xvf mydir.tar.gz
tar -xvf mydir.tar.bz2

↳ tar -xvf mydir.tar
↳ ls # get mydir directory
↳ rm -r mydir # remove mydir
↳ ls # no mydir

↳ tar -xvf mydir.tar.gz # extract unzipping here

↳ tar -xvf mydir.tar.bz2 # extract in mydir

PS and Kill commands → process management

process ⇒ instance of a running program

whenever you run shell command or shell script in unix/linux, a process will be created for that particular command/script.

① PS ⇒ use to see what are all process are running in linux m/c.

② kill ⇒ if some processes are running, & if we want to stop those process/kill then use kill command (to kill specific process)

types of processes

① foreground process ⇒

as soon as executed command, immediately display the o/p

once executed command immediately ~~ones~~ existed from cmd prompt.

eg. pwd

② Background process ⇒

execute same (pwd) command as background process by using & symbol, it giving some process id & it is not giving any cmd prompt to executing other cmd once you press enter it is exited.

eg. pwd &

if u want to execute background process use & symbol

PS ⇒ display process which are currently running on linux/unix mic.

↳ PS

o/p ⇒ process ID terminal time

↳ gedit

↳ cat myscript.sh

↳ sh myscript.sh # process created

ctrl+c ⇒ exit (forcefully exited)
process is not created

↳ PS

sh myscript.sh & # executed as background (background में create होगा)

ctrl+c

↳ PS

created process

↳ PS -f # display complete details
↳ full info about process

↳ PS -e # display all the process which are running in current mic (and internal process)

↳ ps -ef # display all the process with all the user [detail information]

↳ sh myscript.sh & # 1 parent process
1 child process will create (sleep)

PS -f -H # display process in tree structure

↳ ps 2835 # detail info about 2835 process
↑ process ID

Run script as foreground

↳ sh . myscript.sh

ctrl+c (exit)
it will kill the process

↳ ps -f

kill 2851 # delete process / to stop current processes

↳ ps -f # verify in All precess

delete multiple process

↳ sh myscript.sh &
↳ sh myscript.sh &

↳ ps -f

↳ kill 2869, 2871 # delete multiple process

If you want to forcefully kill the process use.

↳ kill -9 2869, 2871

↳ ps -f # verify delete or not (display all process)

↳ ps -f | grep 'myscript.sh'

ps -f → Capture all process → from this process only 'myscript.sh' will display