LINUX OPERATING SYSTEM'S IMPORTANT COMMANDS:

1. KERNEL RELATED COMMANDS:

`# ls -R /lib/modules/$(uname -r)`
To see a list of all the modules currently available on your system, use the following command to list the contents of the `/lib/modules` directory. Linux distros are made up of a staggering number of components, so you should expect a lot of output.

`# modinfo /path/to/module.ko`
Use the following command syntax to display information for a particular module. Of course, replace the name below with the real name of an actual module on your system.

To check Linux Kernel version, try the following commands:

1. **`uname -r`** : Find Linux kernel version
2. **`cat /proc/version`** : Show Linux kernel version with help of a special file
3. **`hostnamectl | grep Kernel`** : For systemd based Linux distro you can use hotnamectl to display hostname and running Linux kernel version.

2. MEMORY MANAGEMENT RELATED COMMANDS:

1. **`$ cat /proc/meminfo`** :
   This is a virtual file that reports the amount of available and used memory. It contains real-time information about the system's memory usage as well as the buffers and shared memory used by the kernel.

2. **`$ grep MemTotal /proc/meminfo:`**
   To get the physical memory from the / proc/meminfo file.

3. **`$ grep VmallocTotal /proc/meminfo:`**
   To get the virtual memory from /proc/meminfo file.

4. **`$ free`** :
   To Display the Amount of Physical and Swap Memory.

5. **`$ vmstat:`** provides general information about processes, memory, paging, block IO, traps, and CPU activity.
6. **`$ top :`** is useful to check memory and CPU usage per process.It displays information about: *uptime, average load, tasks running, number of users logged in, number of CPUs/CPU utilization, memory/swap system processes.*

7. **`$ Dmidecode:`** provides detailed information about our installed system RAM.

3. PROCESS MANAGEMENT RELATED COMMANDS:

Background Processes
If you just add an & sign to any foreground process it will become a background process.

For example:
root@lco-linux-master:~# sleep 500 &
[1] 4676

root@lco-linux-master:~# jobs
[1]+ Running sleep 500 &
Here we are running a sleep command for 500 seconds which helps in delaying the execution and sending that to the background.

To check all the background jobs run the jobs command as follows.
root@lco-linux-master:~# jobs
[1]+ Running sleep 500 &

To bring back a background process to foreground run fg command as follows.
root@lco-linux-master:~# fg

Listing Running Processes
All the running processes can be listed by running ps (process status) command.

```
root@lco-linux-master:~# ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
root      4339  4234  0 16:21 pts/1    00:00:00 -bash
root      4936  4339  0 17:09 pts/1    00:00:00 sleep 1000
root      4937  4339  0 17:09 pts/1    00:00:00 ps -f
```

Let us understand all the parameters of this command.

Column Parameter	Represents
UID	User ID that this process belongs to
PID	Process ID
PPID	Parent process ID (the ID of the process that started it)
C	CPU utilization of process
STIME	Start time of process
TTY	Terminal type associated with the process
TIME	CPU time taken by the process
CMD	The command that started this process

There are few important command arguments which you can use along with `ps` command.

`-a` -> Shows information about all users.
`-x` -> Shows information about processes without terminals.
`-e` -> Displays extended information.
`-u` -> Shows additional information like -f option.

Stopping or Killing Running Processes

There are several ways to stop or kill a process in Linux.

By pressing `ctrl+c`

If a process is running in foreground you can stop or kill that process by pressing `ctrl+c` (the default interrupt character).

For example:

```
root@lco-linux-master:~# sleep 100

^C
root@lco-linux-master:~#
```

By using `kill` command

In order to kill or terminate a process running in background use the `kill` command to kill the process as follows. One should be aware of the process PID to kill it.
`root@lco-linux-master:~# kill -9 5218`

4. I/O MANAGEMENT RELATED COMMANDS:

1) **iostat:**-The iostat command in Linux is used for monitoring system input/output statistics for devices and partitions. It monitors system input/output by observing the time the devices are active in relation to their average transfer rates.
2) **iostat -x Command:** This command shows more details statistics information. iostat command gives I/O devices report utilization as a result.
3) **iostat -d Command:** This command displays only the device report. It is possible to only show the status of the device utilization with the help of -d option.
4) **iostat -xd Command:** This command shows us the extended I/O statistic for device only.
5) **iostat -p Command:** This command display statistics for block devices. With the help of this command, it is possible to directly show information for each block device automatically.
6) **iostat -k Command:** This command captures the statistics in kilobytes or megabytes. By default, iostat measure the I/O system with the bytes unit.
7) **iostat -k 2 3 Command:** This command displays CPU and device statistics with delay.

5. FILE MANAGEMENT RELATED COMMANDS:

1. pwd Command

pwd, short for the print working directory, is a command that prints out the current working directory in a hierarchical order, beginning with the topmost root directory (/).

2. cd Command

To change or navigate directories, use the cd command which is short for change directory.

To go a directory up append two dots or periods in the end.

```
$ cd ..
```

To go back to the home directory run the cd command without any arguments.

```
$ cd
```

3. ls Command

The ls command is a command used for listing existing files or folders in a directory.

4. touch Command

The touch command is used for creating simple files on a Linux system. To create a file, use the syntax:

```
$ touch filename
```

5. cat Command

To view the contents of a file, use the cat command as follows:

```
$ cat filename
```

6. mv Command

To move the file, use the syntax below:

```
$ mv filename /path/to/destination/
```

To rename a file, use the syntax shown. The command removes the original file name and assigns the second argument as the new file name.

```
$ mv filename1 filename2
```

7. cp Command

The cp command, short for copy, copies a file from one file location to another. The syntax for copying a file is shown below.

```
$ cp /file/path /destination/path
```

8. mkdir Command

To create a new directory use the mkdir (make directory) command as follows:

```
$ mkdir directory_name
```

9. rmdir Command

The rmdir command deletes an empty directory. For example, to delete or remove the tutorials directory, run the command:

```
$ rmdir tutorials
```

10. rm Command

The rm (remove) command is used to delete a file. The syntax is quite straightforward:

```
$ rm filename
```

6. SEMAPHORES AND THREADS RELATED COMMANDS:

1. **pthread_create()** - this command is used to create a new thread.The pthread_create() function starts a new thread in the calling process.

2. **pthread_exit()** - This command is used for terminating the calling thread.The pthread_exit() function terminates the calling thread and returns a value via *retval*

3. **pthread_join()** - this command is used to join with a terminated thread. The pthread_join() function waits for the thread specified by *thread* to terminate.The thread specified by *thread* must be joinable. On success, pthread_join() returns 0; on error, it returns an error number.

4. **pthread_cancel()** - this command sends a cancellation request to the thread. Whether and when the target thread reacts to the cancellation request depends on two attributes that are under the control of that thread: its cancelability state and type.

5. **pthread_self()** - obtain ID of the calling thread. The pthread_self() function returns the ID of the calling thread. This is the same value that is returned in **thread* in the *pthread_create* call that created this thread. This function always succeeds, returning the calling thread's ID.

6. **sem_close()** — this command is used to close a named semaphore. The *sem_close*() function shall deallocate any system resources allocated by the system for use by this process for this semaphore.

7. **sem_open()** — this command is used to initialize and open a named semaphore. The *sem_open*() function shall establish a connection between a named semaphore and a process.

8. **sem_getvalue()** — this command is used to get the value of a semaphore. The *sem_getvalue*() function shall update the location referenced by the *sval* argument to have the value of the semaphore referenced by *sem* without affecting the state of the semaphore.

7. NETWORK SYSTEM RELATED COMMANDS:

1.ifconfig:

Linux ifconfig stands for interface configurator.ifconfig is used to initialize an interface, configure it with an IP address, and enable or disable it.

2. ip

This is the latest and updated version of the ifconfig command.

Syntax:

1. ip a
2. ip addr

This command gives the details of all networks like ifconfig.This command can also be used to get the details of a specific interface.

3.traceroute

. It is used to troubleshoot the network. It detects the delay and determines the pathway to your target.

4.tracepath

It traces the route to the specified destination and identifies each hop in it. If your network is weak, it recognizes the point where the network is weak.

5.ping

Linux ping is one of the most used network troubleshooting commands. It basically checks for the network connectivity between two nodes.ping stands for Packet INternet Groper.

6.ss

Linux ss command is the replacement for netstat command. It is regarded as a much faster and more informative command than netstat.The faster response of ss is possible as it fetches all the information from within the kernel userspace.

7.dig

Linux dig command stands for Domain Information Groper. This command is used in DNS lookup to query the DNS name server. It is also used to troubleshoot DNS related issues.It is mainly used to verify DNS mappings, MX Records, host addresses, and all other DNS records for a better understanding of the DNS topography.

8.route

Linux route command displays and manipulates the routing table existing for your system.A router is basically used to find the best way to send the packets across to a destination.

9.curl & wget

Linux curl and wget commands are used in downloading files from the internet through CLI. The curl command has to be used with the option "O" to fetch the file, while the wget command is used directly.

10.tcpdump

Linux tcpdump command is the most used command in network analysis among other Linux network commands. It captures the traffic that is passing through the network interface and displays it.

9. DEVICE DRIVERS RELATED COMMANDS:

To list all the device files use the below command.

```
ls -l /dev
```

In the above output, we can see some other types of file types, some of them have B for a block device, C for character device, some devices start with /dev/sda or /sdb. In Linux, the disk names are alphabetical. For example, dev/sda is the first hard drive, dev/sdb is the second hard drive, and so on. These devices are mass storage devices like memory sticks, hard drives, etc. Hence, sda means that this device was detected by the computer first. Examples of character devices are : */dev/consoles* or */dev/ttyS0*. These devices are accessed as a stream of bytes. Example of block device: /*dev/sdxn*. Block devices allow the programmer to read and write any size of the block. Pseudo devices act as device drivers without an actual device. Examples of pseudo devices are */dev/null, /dev/zero, /dev/pf* etc.

1. fdisk – It stands for format disk. This command is used to display the partitions on a disk and other details related to the file system.

```
sudo fdisk -l
```

2. sfdisk – This command displays the partitions on the disk, the size of each partition in MB, etc.

3. parted – This command helps list and modify the partitions of the disk.

```
sudo parted -l
```

4. df – Displays the details of the file system. Using grep we can filter real hard disk files.

```
df -h | grep ^/dev
```

5. lsblk – List details about the block devices.

```
lsblk
```

6. inxi – Lists details about the hardware components in the file system.

```
inxi -D -xx
```

The kernel is a monolithic piece of software, but it is also responsible to provide support to the hardware. Most of the devices have built-in kernel modules, so when they are plugged in, they start working automatically.