**1. Ubuntu vs Windows – Key Differences**

- **Cost:** Ubuntu free, Windows paid.
- **Security**: Ubuntu safer, Windows more virus-prone.
- **Performance:** Ubuntu lighter, Windows heavier.
- **Software:** Windows supports more apps/games, Ubuntu limited.
- **User Interface:**
  - Ubuntu → GNOME (customizable, less beginner-friendly).
  - Windows → Familiar GUI, user-friendly.
- **Customization:** Ubuntu highly customizable, Windows less.

**2. What is karnel?**

A kernel is the core part of an operating system.

- It manages hardware and software communication.
- Controls CPU, memory, devices, and processes.
- Acts as a bridge between applications and hardware.

👉 In short: Kernel = the heart of the OS that controls everything.

**3. Kernel vs OS vs Distro**

- **Kernel** → Core part of OS, manages hardware & resources. (e.g., Linux kernel)
- **Operating System (OS)** → Full system that includes kernel + tools + UI to run a computer. (e.g., Linux, Windows)
- **Distribution (Distro)** → A packaged version of an OS (kernel + software + desktop environment). (e.g., Ubuntu, Fedora)

👉 Simple: **Kernel = heart, OS = body, Distro = different flavors of that body.**

**4. Ubuntu Variants: Desktop, Server, Minimal**

**Ubuntu Variants**:

- **Ubuntu Desktop** → For personal computers; comes with GUI (GNOME).
- **Ubuntu Server** → For servers; no GUI by default, optimized for performance and services.

- **Ubuntu Minimal** → Lightweight version; only core packages, for custom setups.

👉 Simple: **Desktop = GUI use, Server = services, Minimal = lightweight/custom.**

**5. UEFI, BIOS, GRUB Bootloader**

- **BIOS (Basic Input/Output System)** → Firmware that starts the computer and loads the bootloader.
- **UEFI (Unified Extensible Firmware Interface)** → Modern firmware that replaces BIOS, faster, supports large disks, secure boot.
- **GRUB (GRand Unified Bootloader)** → Program that loads and manages the operating system after BIOS/UEFI (Bootloader).

  Example: GRUB, Windows, Boot Manager.

👉 Simple: **BIOS/UEFI = starts system, GRUB = loads OS, Bootloader = starter that launches the OS.**

👉 Flow: **UEFI/BIOS → GRUB → OS**.

**6. Filesystem Types: ext4, swap, NTFS**

- **ext4** (Fourth Extended Filesystem) → Default Linux filesystem, reliable, supports large files.
- **swap** → Disk space used as extra RAM.
- **NTFS** (New Technology File System**)** → Default Windows filesystem, supports permissions, large files.

👉 Simple: **ext4 = Linux data, swap = virtual RAM, NTFS = Windows data.**

# 50 Essential Linux Terminal Commands

## A. System Navigation

1. `pwd` → Prints the current working directory.
2. `ls -l` → Lists files in long format (permissions, owner, size, date).
3. `cd` → Changes the current directory.
4. `tree` → Displays directory structure in a tree-like format.
5. `find / -name filename` → Searches for a file by name starting from `/`.
6. `du -sh *` → Shows disk usage of files/folders in human-readable format.
7. `df -h` → Shows free and used disk space on mounted filesystems.
8. `stat filename` → Displays detailed file info (size, permissions, timestamps).
9. `realpath file` → Prints the absolute path of a file.
10. `basename path` → Extracts the filename from a full path.

---

## B. File Management

11. `touch file.txt` → Creates an empty file or updates its timestamp.
12. `cp file1 file2` → Copies `file1` to `file2`.
13. `mv file1 newname` → Moves/renames a file.
14. `rm file.txt` → Deletes a file.
15. `mkdir dir` → Creates a new directory.
16. `rmdir dir` → Removes an empty directory.
17. `nano file.txt` → Opens a file in the Nano text editor.
18. `cat file` → Displays file contents.
19. `more file` → Views file contents page by page (forward only).
20. `less file` → Views file contents with navigation (forward & backward).

---

## C. Permissions & Ownership

- `chmod +x script.sh` → Makes a script/executable file runnable by giving execute permission.
- `chown user:group file` → Changes ownership of a file to a specific user and group.

- **`ls -lah`** → Lists files in a directory with detailed info, human-readable sizes, and hidden files.
- **`umask`** → Shows/sets default file permission mask for newly created files.
- **`id`** → Displays user ID (UID), group ID (GID), and group memberships of the current user.

---

## D. Process Management

- **`ps aux`** → Shows all running processes with detailed information.
- **`top`** → Real-time view of running processes, resource usage, and system load.
- **`htop`** → Improved interactive version of `top` (install separately).
- **`kill PID`** → Terminates a process using its Process ID.
- **`pkill processname`** → Kills processes by name instead of PID.
- **`jobs`** → Lists background and stopped jobs in the current shell.
- **`fg`** → Brings a background job to the foreground.
- **`bg`** → Resumes a stopped job in the background.

---

## E. Networking

- **`ip a`** → Shows all network interfaces, IP addresses, and status.
- **`ping google.com`** → Tests connectivity to a host by sending ICMP packets.
- **`ifconfig`** → Displays/sets network interface configuration (from `net-tools`).
- **`netstat -tulnp`** → Lists open ports, active connections, and listening services.
- **`curl example.com`** → Fetches data from a URL (supports many protocols).
- **`wget http://file`** → Downloads files from the web.
- **`hostname`** → Shows or sets the system's hostname.
- **`nmap`** → Advanced network scanner for discovering hosts and services.

### 1. `lsblk` (List Block Devices)

- Shows **all block devices** (disks, partitions, USB drives, etc.).
- Displays structure like a **tree view**.
- Doesn't show free space — just device layout.

Example:

```
lsblk
```

Output:

```
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0  100G  0 disk
├─sda1   8:1    0   96G  0 part /
└─sda2   8:2    0    4G  0 part [SWAP]
sdb      8:16   0   32G  0 disk /media/usb
```

- `sda` → Main disk.
- `sda1` → Root partition `/`.
- `sda2` → Swap partition.
- `sdb` → USB drive.

---

### 2. `df -h` (Disk Free - human readable)

- Shows **disk usage (used/free space)** of mounted filesystems.
- Units shown in **human-readable format** (MB, GB).

Example:

```
df -h
```

Output:

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        96G   20G   71G  22% /
/dev/sdb1        32G    5G   25G  17% /media/usb
tmpfs           2.0G     0  2.0G   0% /dev/shm
```

- `Size` → Total size.
- `Used` → Space used.
- `Avail` → Free space left.
- `Mounted on` → Where it's attached in the filesystem.

---

✅ Difference at a Glance

| Command | Purpose | Shows |
|---|---|---|
| `lsblk` | Device layout | Disks, partitions, mount points (no usage %) |
| `df -h` | Disk usage | Size, used, available, usage %, mount points |

**1. Create a folder, move into it, and create 5 dummy files**

```
mkdir linux_lab
cd linux_lab
touch file1.txt file2.txt file3.txt file4.txt file5.txt
```

---

**2. Redirect the output of `ls -l` into list.txt**

```
ls -l > list.txt
```

---

**3. Command chaining example**

```
mkdir test && cd test && touch hello.txt
```

---

**4. Find all `.txt` files in /home**

```
find /home -name "*.txt"
```

---

**5. Short notes on practice commands**

- **Ctrl+R** → Search command history.
- **!!** → Run the last command again.
- **Tab** → Auto-complete file/directory names.

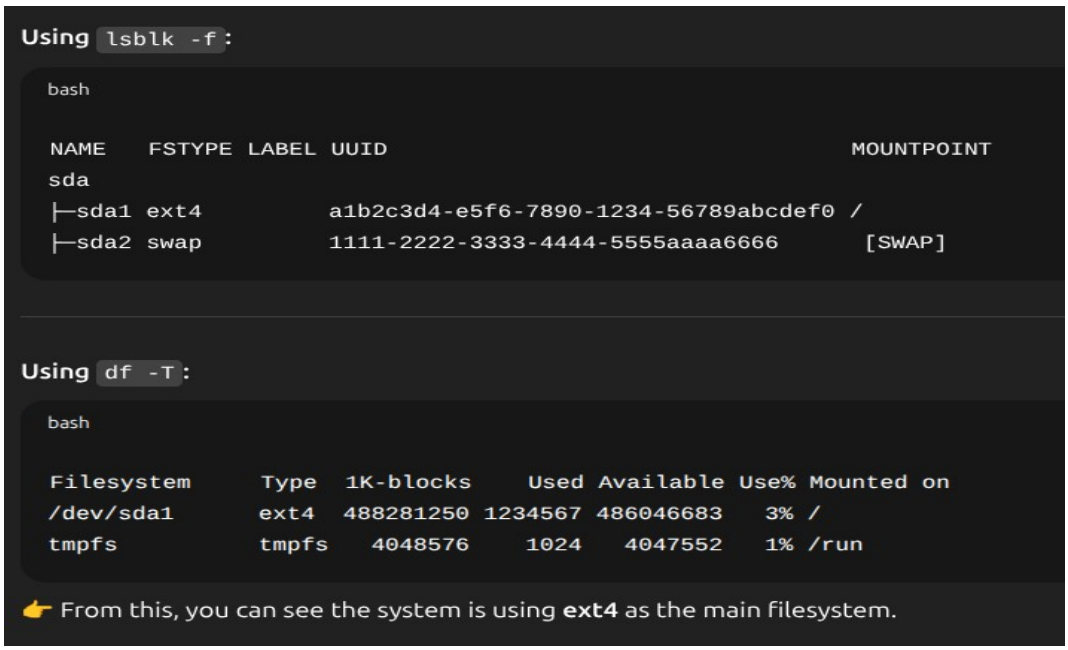
**1. Key Directories**

- **/home** → Stores user files and personal data.
- **/etc** → Configuration files for the system.
- **/var** → Variable data like logs, mail, cache.
- **/bin** → Essential user commands (binaries).

---

**2. Find current filesystem**

👉 **Both show the filesystem type (e.g., ext4, xfs).**



**3. Navigate to /etc directory,list all hidden file, return back to home**

**Here's the short solution with commands:**

```
cd /etc          # go to /etc directory
ls -a            # list all files including hidden ones
cd ~             # return back to home directory
```

👉 `-a` option shows hidden files (those starting with `.`).

**4. Write a command that redirect both output and error of the command ls /nonexistent into a file named error.log**

ls /nonexistent &> error.log

**5. File permissioning**

-rwx--x--x 1 ashfak developers 0 Sep 10 21:00 script.sh

**Columns:**

`rwx` → owner can read, write, execute

`--x` → group can execute only

`--x` → others can execute only

1 → Number of links

ashfak → Owner (user)

developers → Group

0 → File size in bytes

Sep 10 21:00 → Last modified date & time

script.sh → Filename

👉 Simple: ls -l shows permissions, ownership, size, date, and name.

**6. Change the permission of script.sh**

*chmod 711 script.sh*

Explanation:

7 (owner) → read + write + execute

1 (group) → execute only

1 (others) → execute only