# WIPRO NGA Program – DC DWS Batch 7

Capstone Project Presentation – 4th and 5th Sept 2024

Project Title Here - FILE SYSTEM , LVM in LINUX

Presented by - MUKESH YADAV

# FILE SYSTEM IN LINUX

*BY MUKESH YADAV*

BATCH NAME : DATA CENTRE - BATCH 7

CLOUD USERNAME : 24NAG2180_U07

GIT USERNAME : b166user05

# Contents

- File System In Linux
- Objectives
- Project Scope
- Prerequisites , Project Requirements
- Deliverables

# What is Linux ?

Linux is an  open-source operating system based on Unix. Known for its stability, security and flexibility. It’s free to use, supports multiple users and processes , and is highly customizable, fitting various environments from desktops to servers and embedded systems.

# What is the purpose of this project on Linux file systems and LVM ?

The purpose of the project is to teach students about the Linux file system architecture, how to manage disk partitions, and how to use Logical Volume Management (LVM) to create, resize, and manage storage volumes efficiently.

# How do you manage disk partitions and file systems in Linux?

Disk partitions are managed using tools like **‘parted’**and **‘fdisk’** for creating, resizing, and deleting partitions. File systems are created and formatted using the **‘mkfs’** command, and they are mounted or unmounted using the **‘mount’** and **‘umount’** commands.

## What are the key components and benefits of Logical Volume Management (LVM) ?

LVM consists of Physical Volumes (PVs), Volume Groups (VGs), and Logical Volumes (LVs). It allows for flexible and dynamic disk management, such as resizing volumes and adding new disks without disrupting the system.

# OBJECTIVES

1. Understand the file System Architecture in Linux Operating system.
2. Create the partitions and mount the file systems.
3. Manage the file system and ensure enough disk space is available.
4. Edit the necessary file and make the file system mount while booting.
5. How to create and delete Disk using “Parted” command in Linux.

# File System Architecture in Linux

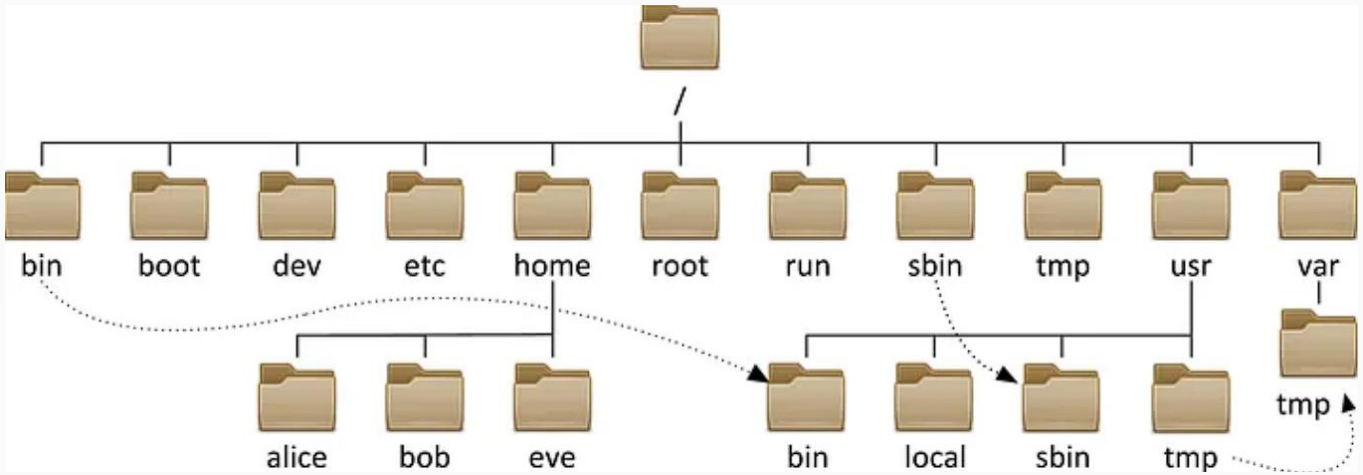
**File system Hierarchy :** The linux file system starts from the root directory (/) and includes key subdirectories like /home for user files, /etc for configurations, and /var for variable data.

**File Types :** Files are categorized into regular files , directories, symbolic links and special files.

**File Permissions :** Manage file access with commands such as **‘chmod’**, **‘chown’**, and ‘**chrgrp**’.

**Common File Systems :** Ext4, xfs and btrfs are popular Linux file systems, each offering distinct features for various use cases.

# File system Hierarchy




The Linux file system starts at the root directory (`/`), which is the base of the file structure. From there, it branches into important subdirectories:

- **`/home`:** This is where user-specific files and personal data are stored. Each user has their own directory under `/home`.
- **`/etc`:** Contains system-wide configuration files that control the operating system and installed software.
- **`/var`:** Used for variable data that changes frequently, such as system logs, mail spools, and temporary files.

# File Types

```
[root@localhost ~]# pwd
/root
[root@localhost ~]# ls
anaconda-ks.cfg
[root@localhost ~]# touch test1.txt
[root@localhost ~]# mkdir test
[root@localhost ~]# ln -s test1.txt   test2.txt
[root@localhost ~]# mknod /dev/block_device b  8 16
[root@localhost ~]# mknod /dev/char_device c  8 16
[root@localhost ~]# ls -l /dev/char_device
crw-r--r--. 1 root root 8, 16 Aug 31 17:46 /dev/char_device
[root@localhost ~]# ls -l /dev/block_device
brw-r--r--. 1 root root 8, 16 Aug 31 17:45 /dev/block_device
[root@localhost ~]# ls -l test1.txt
-rw-r--r--. 1 root root 0 Aug 31 17:32 test1.txt
[root@localhost ~]#
[root@localhost ~]# ls -l test2.txt
lrwxrwxrwx. 1 root root 9 Aug 31 17:33 test2.txt -> test1.txt
[root@localhost ~]# ls -la test
total 4
drwxr-xr-x. 2 root root    6 Aug 31 17:32 .
dr-xr-x---. 6 root root 4096 Aug 31 17:45 ..
```

Files are categorized into regular files, directories, symbolic links, character and block device files

- **touch:** Create an empty file or update the timestamp of an existing file.
- **mkdir:** Create a new directory.
- **mknod:** Create a special file (like a device file).
- **ln -s:** Create a symbolic (soft) link to a file or directory, which points to the original file or directory without duplicating its content.
- **ls -l:** List files in detail with permissions, ownership, and timestamps.

# File Permissions

```
[root@localhost ~]# touch test5.txt
[root@localhost ~]# ls -l test5.txt
-rw-r--r--. 1 root root 0 Aug 31 18:04 test5.txt
[root@localhost ~]# chmod u=rwx,g=rw,o=rw    test5.txt
[root@localhost ~]# ls -l test5.txt
-rwxrw-rw-. 1 root root 0 Aug 31 18:04 test5.txt
[root@localhost ~]# chown User4 test5.txt
[root@localhost ~]# chgrp group1 test5.txt
[root@localhost ~]# ls -l test5.txt
-rwxrw-rw-. 1 User4 group1 0 Aug 31 18:04 test5.txt
[root@localhost ~]#
```

**File Permissions:** Control who can read, write, or execute files with commands:

- **chmod:** Change file permissions (e.g., read, write, execute).
- **chown:** Change file ownership (user and/or group).
- **chgrp:** Change the group ownership of a file.

# Common File Systems

```
[root@localhost ~]# lsblk -f
NAME            FSTYPE      LABEL UUID                                   MOUNTPOINT
sda
├─sda1          xfs               97e92bf8-6b30-4260-9418-31a22fc40b49   /boot
└─sda2          LVM2_member       BSJqvd-RjIz-oSjb-otcJ-3Ebh-Uj2G-EBWa7O
  ├─centos-root xfs               7f20de15-9d76-4f74-8f46-5f1a3abd486f   /
  └─centos-swap swap              4f46c356-6dc8-405d-932b-879cc767cadc   [SWAP]
sdb
└─sdb1          ext4              be6aa3b0-97fc-48a5-81a6-98c69f8ea1af
sdc
sr0
```

- **Ext4:** Reliable and widely used, offering solid performance for general-purpose needs.
- **XFS:** High-performance and scalable, ideal for handling large files and enterprise systems.
- **`lsblk -f`:** Shows information about block devices and their file system types, aiding in storage management.

Manage the file system and ensure the enough disk space is available.

**Monitoring and Managing Disk space :** Use **df** to check disk space usage and du to to report file and directory sizes

**Clearing and Organizing Disk Space :** Use rm to delete files or directories and **rmdir** to remove empty directories.

# Monitoring and Managing Disk space using (df)

```
[root@localhost ~]# df
Filesystem              1K-blocks    Used Available Use% Mounted on
devtmpfs                   914380       0    914380   0% /dev
tmpfs                      931484       0    931484   0% /dev/shm
tmpfs                      931484   10624    920860   2% /run
tmpfs                      931484       0    931484   0% /sys/fs/cgroup
/dev/mapper/centos-root  17811456 4522968  13288488  26% /
/dev/sda1                 1038336  191176    847160  19% /boot
tmpfs                      186300      24    186276   1% /run/user/1000
[root@localhost ~]# df -h
Filesystem              Size  Used Avail Use% Mounted on
devtmpfs                893M     0  893M   0% /dev
tmpfs                   910M     0  910M   0% /dev/shm
tmpfs                   910M   11M  900M   2% /run
tmpfs                   910M     0  910M   0% /sys/fs/cgroup
/dev/mapper/centos-root   17G  4.4G   13G  26% /
/dev/sda1               1014M  187M  828M  19% /boot
tmpfs                    182M   24K  182M   1% /run/user/1000
```

- **df:** Check disk space usage to see how much space is used and available on mounted filesystems.
- **df -h:** Check disk space usage in a human-readable format, showing sizes in KB, MB, or GB for easier understanding.

# Monitoring and Managing Disk space using (du)

```
[root@localhost ~]# du
4       ./.cache/dconf
4       ./.cache/abrt
8       ./.cache
4       ./.dbus/session-bus
4       ./.dbus
0       ./.config/abrt
0       ./.config
0       ./test
68      .
[root@localhost ~]# du -h
4.0K    ./.cache/dconf
4.0K    ./.cache/abrt
8.0K    ./.cache
4.0K    ./.dbus/session-bus
4.0K    ./.dbus
0       ./.config/abrt
0       ./.config
0       ./test
68K     .
```

- **du:** Report disk usage of files and directories, showing how much space they occupy.
- **du -h:** Report disk usage in a human-readable format, displaying sizes in KB, MB, or GB for easier understanding.

# Clearing and Organizing Disk Space

```
[root@localhost ~]# ls
anaconda-ks.cfg  test  test1.txt  test2.txt  test4.txt  test5.txt  test.txt
[root@localhost ~]# rm test1.txt
rm: remove regular empty file 'test1.txt'? y
[root@localhost ~]# rmdir test
[root@localhost ~]# ls
anaconda-ks.cfg  test2.txt  test4.txt  test5.txt  test.txt
[root@localhost ~]# du -h
4.0K    ./.cache/dconf
4.0K    ./.cache/abrt
8.0K    ./.cache
4.0K    ./.dbus/session-bus
4.0K    ./.dbus
0       ./.config/abrt
0       ./.config
68K     .
[root@localhost ~]#
```

- **rm:** Permanently delete files or directories; cannot be undone, so use with caution to avoid accidental data loss.
- **rmdir:** Remove empty directories; does not work if the directory contains files or other directories.

# Edit the necessary file and make the file system while booting.

- **Configuring Automatic File System Mounting** : To ensure a file system mounts automatically at boot, you need to edit the `/etc/fstab` file, identify UUIDs using `blkid`, and verify the configuration with `mount -a`.

# Configuring Automatic File System Mounting

```
[root@localhost ~]# blkid  /dev/sdb1
/dev/sdb1: UUID="49f30a3b-6d6a-4626-8cc3-84a2743a3453" TYPE="ext4"
[root@localhost ~]# vi /etc/fstab
```

File Edit View Search Terminal Help

```
#
# /etc/fstab
# Created by anaconda on Fri Aug 30 20:19:56 2024
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=97e92bf8-6b30-4260-9418-31a22fc40b49 /boot                   xfs     defaul
ts        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
UUID=49f30a3b-6d6a-4626-8cc3-84a2743a3453       /part1  ext4    defaults
0 0
~
:wq!
[root@localhost ~]# mount -a
```

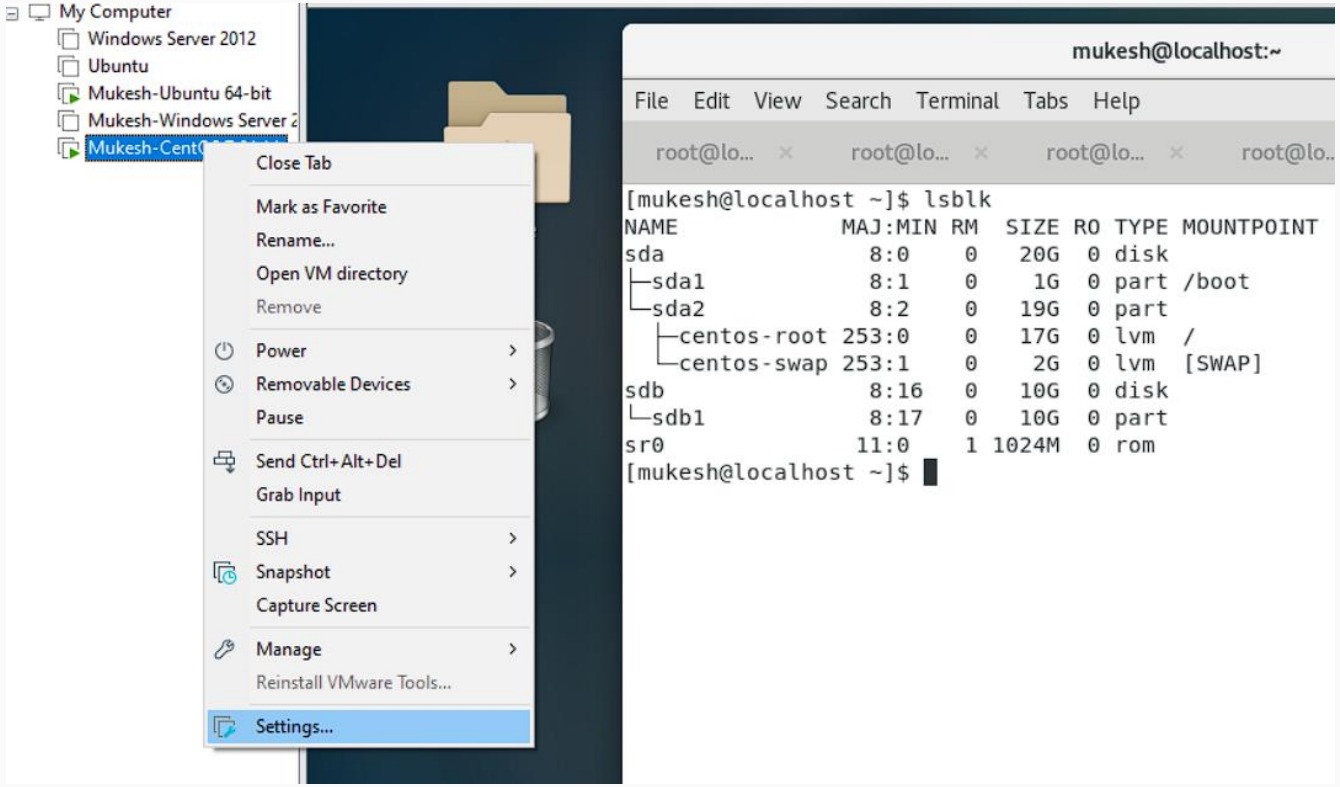
- **`blkid`:** It is used to identify the UUIDs of file systems, which are needed for accurate configuration.
- **`vi /etc/fstab`:** It is used to add or update entries for automatic mounting, with `wq` to save and exit. Finally,
- **`mount -a` :** By this command we can test and apply the new configuration to ensure all listed file systems mount correctly at boot.

# How to create and Delete a Disk Partition in Linux using the parted Command.

## Disk Partition Management Overview

- **Create a Partition:** Use `parted` to create a new disk partition, defining size and type to meet your needs.
- **Mount the Partition:** Mount the newly created partition to a directory so it can be accessed and used by the system.
- **Update `/etc/fstab`:** Modify the `/etc/fstab` file to ensure the new partition mounts automatically at boot.
- **Resize the Partition:** Use `parted` to adjust the size of the partition as needed to fit your storage requirements.
- **Delete the Partition:** Remove the partition using `parted` when it's no longer needed, ensuring any data is backed up beforehand.
- **Update `/etc/fstab` Again:** Modify `/etc/fstab` to remove references to the deleted partition and avoid boot errors.

# Step 1




**1. Check Existing Block Devices:**

- Use the `**lsblk**` command to list all block devices and verify if there is an existing empty disk available.

**2. Proceed Based on Disk Availability**:

- If an empty disk is available, you can skip the manual disk creation and proceed directly to using the `**parted**` command.

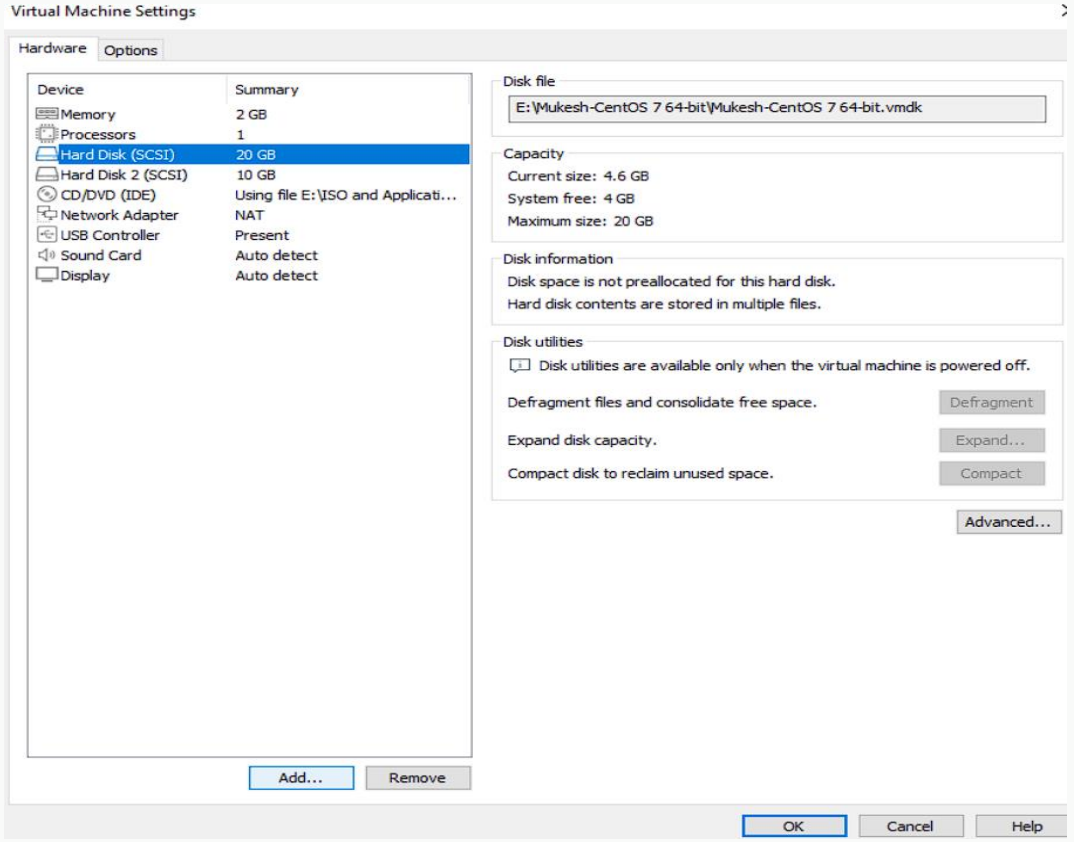
- If no empty disk is available, you need to create a new disk manually.

**3. Create a New Disk Manually:**

- Right-click on the OS machine named "User_centos."

- Click on **"Settings"** to access the configuration options for adding a new disk.

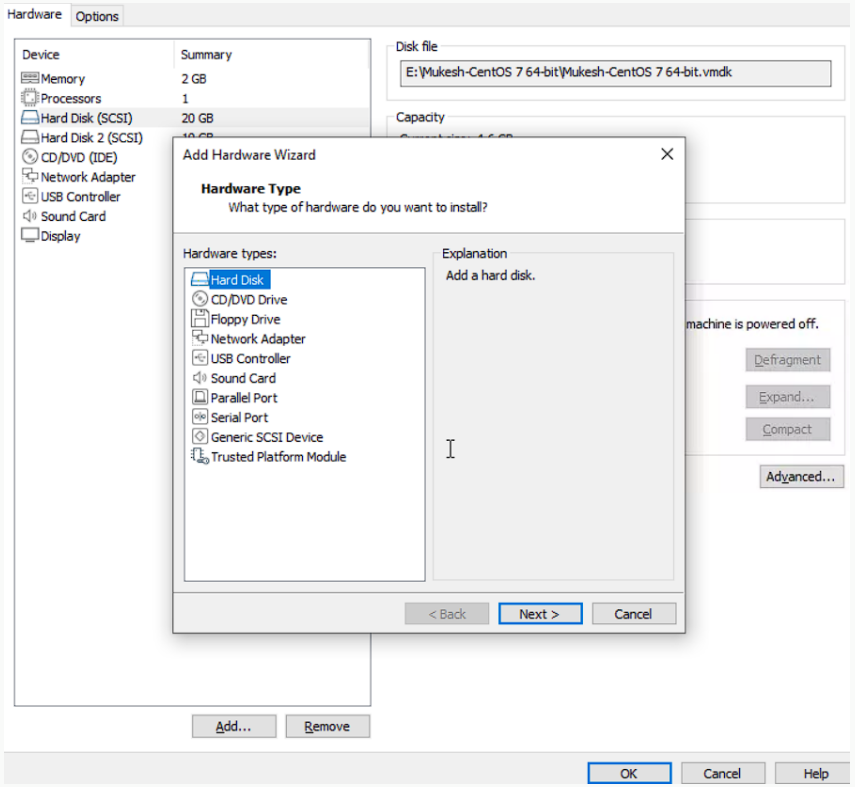
# Step 2




**1.** Click on the **"Hard disk"** option.

**2.** Click on the **"Add"** button below to proceed with the next steps.

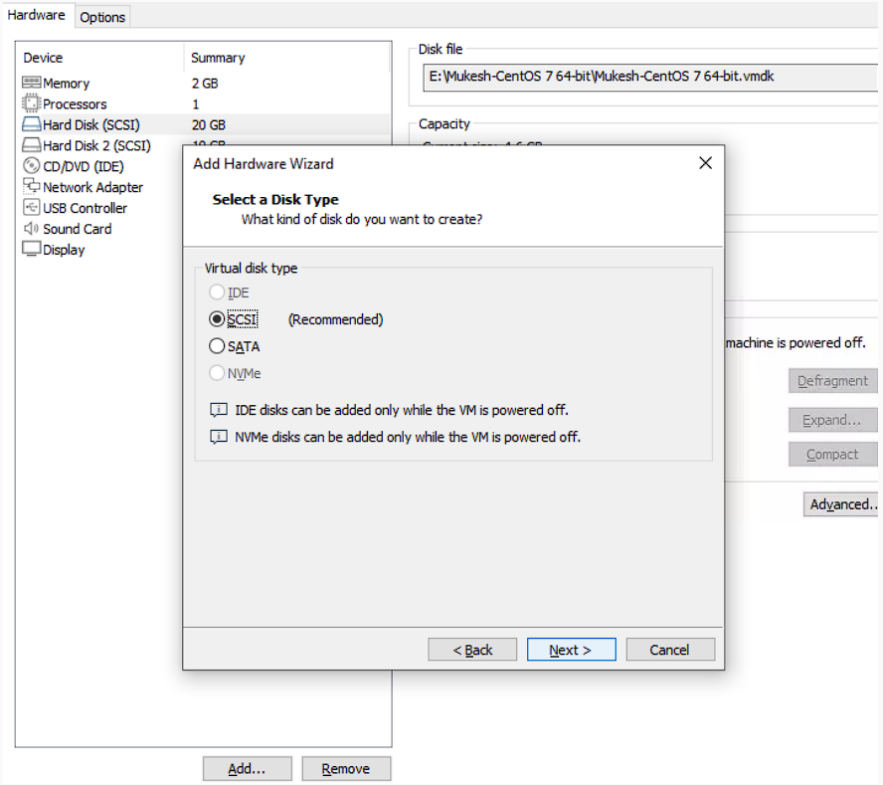
# Step 3




**1.** Click on the **"Hard disk"** option.

**2.** Click on **"Next"** to proceed with the next steps.

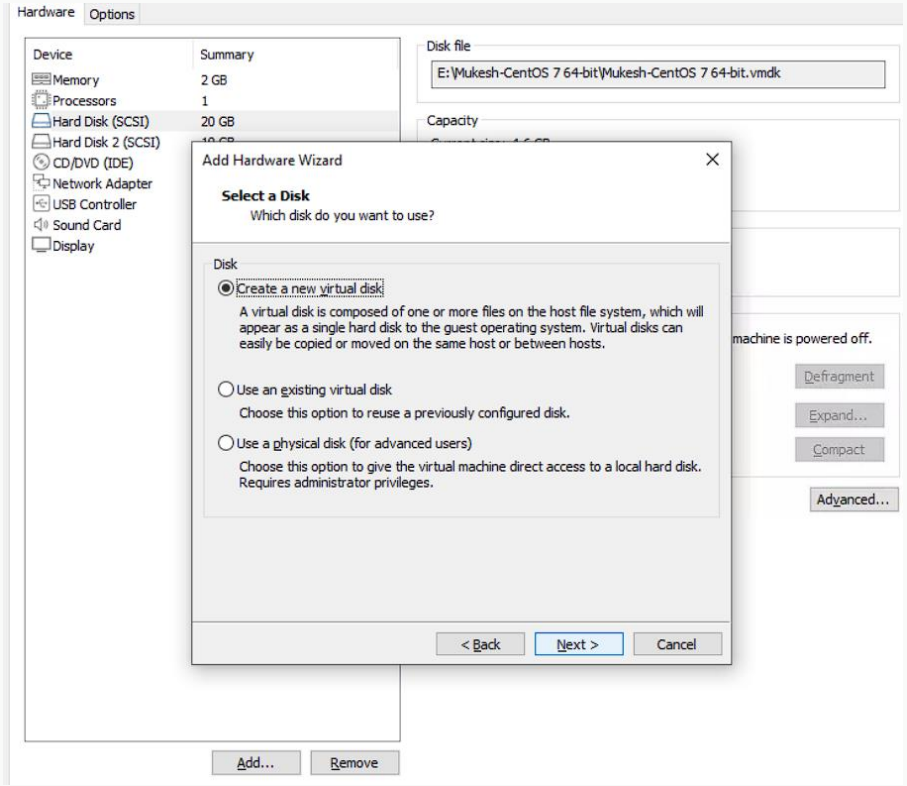
# Step 4




**1.** Select the **"SCSI"** option, which is already recommended.

**2.** Click on **"Next"** to proceed with the next steps.

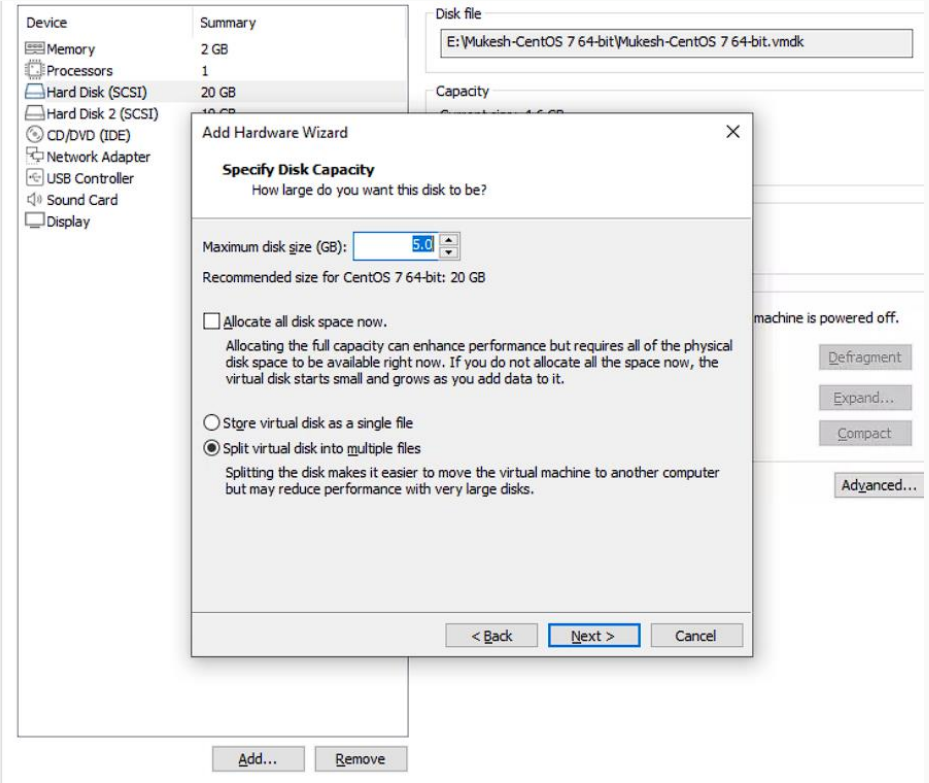
# Step 5




**1.** Select the **“Create new virtual disk"** option, which is already recommended.

**2.** Click on **"Next"** to proceed with the next steps.

# Step 6

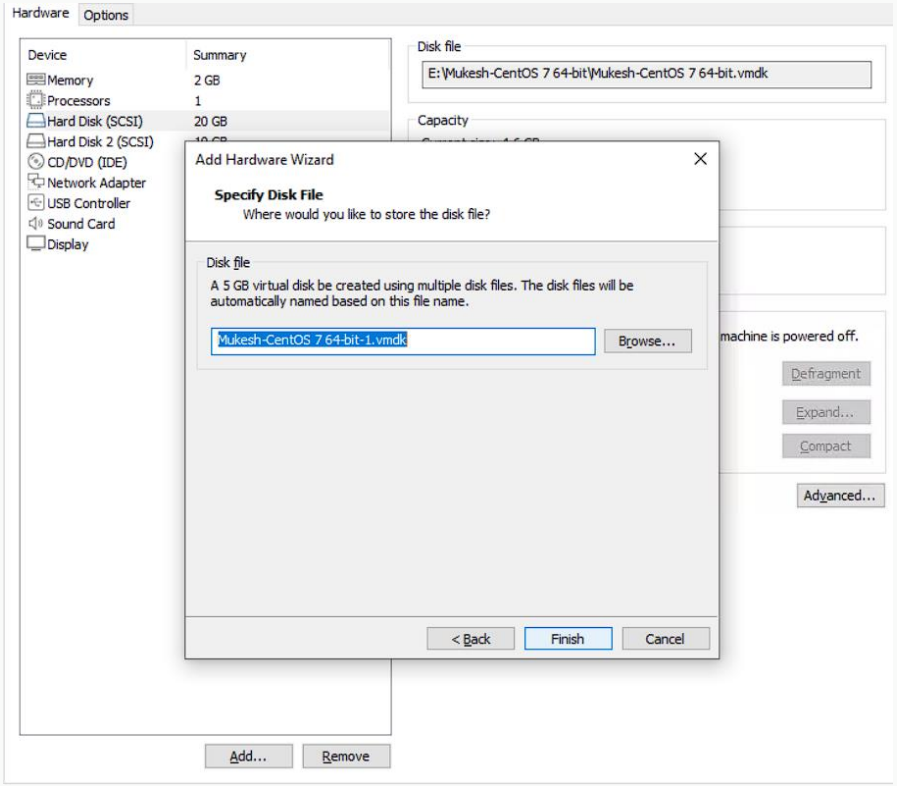



**1.** Select the **"Split virtual disk into multiple files"** option, which is already recommended.

**2.** Enter the **disk size,** I entered the "10 GB."

**3.** Click on **"Next"** to proceed with the next steps.

# Step 7

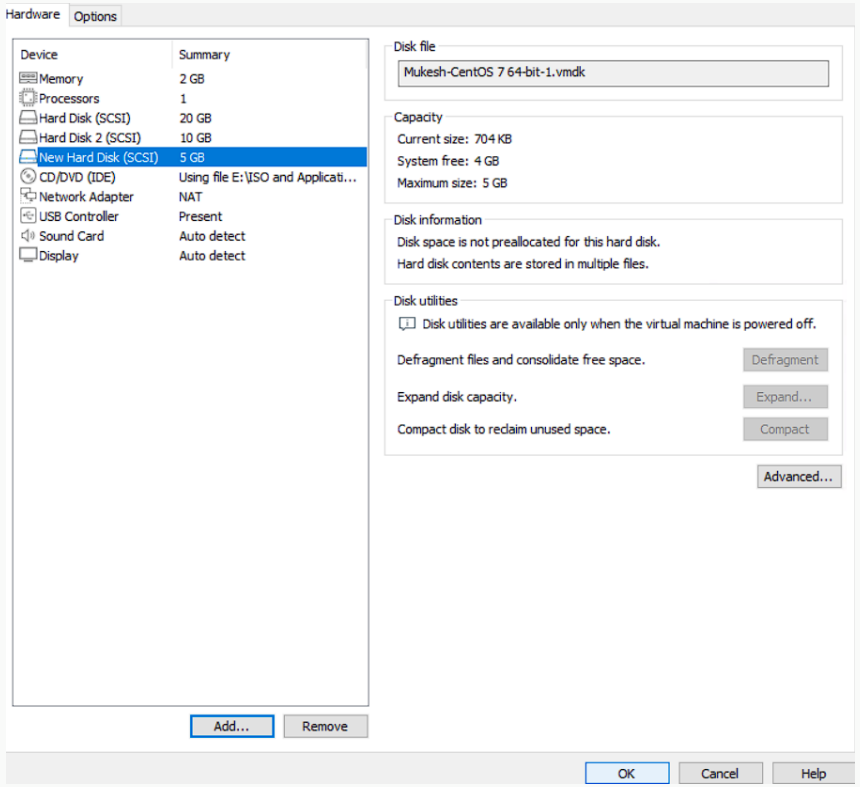



1. Leave the file name as it is.

2. Click on **"Finish"** to complete the creation of the virtual disk.

3. Wait for the next step.

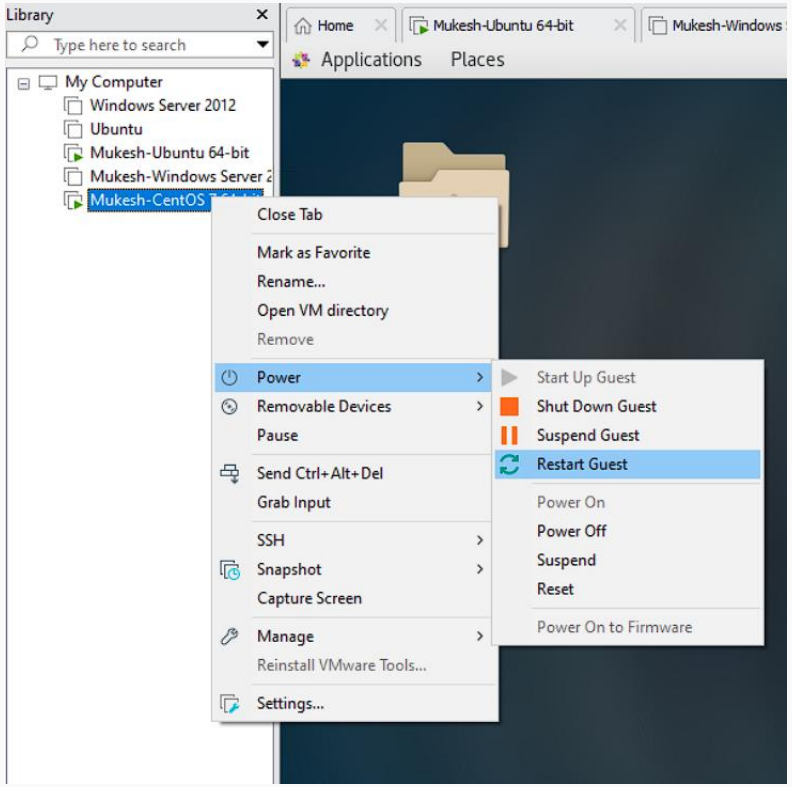
# Step 8




1. Click on **"OK."**

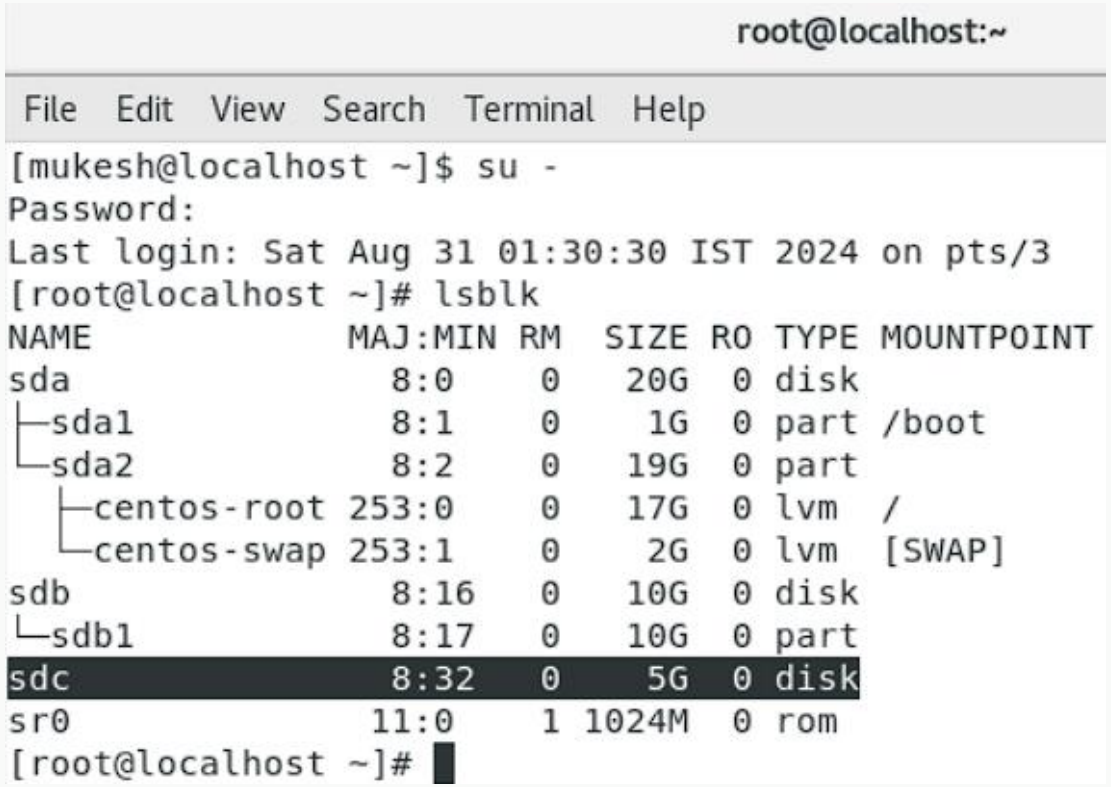
2. After clicking "OK," wait for the next step.

# Step 9




- We created the disk, but the changes will take effect when we reboot the system
- Right-click on the OS machine named **"centos."**
- Click on the **"Power"** option.
- Select **"Restart Guest"** to reboot the system and apply the changes.

# Step 10




**1. Open a Terminal:**

- Open a terminal in the OS (CentOS in this case).

**2. Switch to Root User:**

- Use the command `**su -**` to switch to the root user.

- Enter the password when prompted.

**3. Verify Disk Creation:**

- After switching to the root user, run the command `**lsblk**` to check if the disk was created.

- Verify the disk , Here It appears as **"sdc"** with the path `/dev/sdc`.

# Step 11

```
[root@localhost ~]# parted /dev/sdc
GNU Parted 3.1
Using /dev/sdc
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mktable gpt
(parted) mkpart
Partition name?  []? p1
File system type?  [ext2]? ext4
Start? 1G
End? 2G
(parted) p
Model: VMware, VMware Virtual S (scsi)
Disk /dev/sdc: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name  Flags
 1      1000MB  2000MB  999MB                p1

(parted) q
Information: You may need to update /etc/fstab.
```

**1. Create a Partition with `parted`:** Run the command **`parted /dev/sdc`** to start partitioning the disk.

**2. Create a GPT Table:** Use the command **`mktable gpt`** to create a GPT partition table on the disk.

**3. Create a New Partition:**

Use the command **`mkpart`** to create a **partition**, **name** it "p1," set the **file system type** to "ext4," specify **a start size** of "1G" and an **end size** of "2G," and press **Enter** to apply the settings.

**4. Review and Exit:** Use the command **`p`** to print the partition changes and type **`q`** to quit the `parted` prompt.

# Step 12

```
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
sdc               8:32   0    5G  0 disk
└─sdc1            8:33   0  953M  0 part
sr0              11:0    1 1024M  0 rom
```

I used `**lsblk**` to check if the partition was created. The partition appeared as **"sdc1"** with the path **"/dev/sdc1."**

# Step 13

```
[root@localhost ~]# mkfs.ext4  /dev/sdc1
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
61056 inodes, 243968 blocks
12198 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=251658240
8 block groups
32768 blocks per group, 32768 fragments per group
7632 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: 0/8


done
```

I used `**mkfs.ext4 /dev/sdc1**` to format the disk `**/dev/sdc1**` with the EXT4 file system.

# Step 14

```
[root@localhost ~]# udevadm settle
[root@localhost ~]# partprobe  /dev/sdc
```

```
[root@localhost ~]# parted /dev/sdc
GNU Parted 3.1
Using /dev/sdc
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) p
Model: VMware, VMware Virtual S (scsi)
Disk /dev/sdc: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size   File system  Name  Flags
 1      1000MB  2000MB  999MB  ext4         p1

(parted) resizepart  1  3GiB
(parted) p
Model: VMware, VMware Virtual S (scsi)
Disk /dev/sdc: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name  Flags
 1      1000MB  3221MB  2221MB  ext4         p1
```

I used `**udevadm settle**` and `**partprobe /dev/sdc**` to update the system's partition table, then used `**parted /dev/sdc**` to modify the partition; to resize a partition to 1 GB from a previous 2 GB size, you need to specify the new total size, including the old size, as `**3GiB**` to encompass the previous partition size and selected **partition 1 ( resizepart 1 3GiB).**

# Step 15

```
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
sdc               8:32   0    5G  0 disk
└─sdc1            8:33   0  2.1G  0 part
[root@localhost ~]# mkdir  /part2
[root@localhost ~]# mount  /dev/sdc1  /part2
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
sdc               8:32   0    5G  0 disk
└─sdc1            8:33   0  2.1G  0 part /part2
```

1. Used `**lsblk**` to verify the partition size.

2. Created a directory with `**mkdir /part2**`.

3. Mounted `**/dev/sdc1**` to `**/part2**` using `**mount /dev/sdc1 /part2**`.

4. Used `**lsblk**` again to confirm that the partition was mounted correctly.

# Step 16

```
[root@localhost ~]# blkid /dev/sdc1
/dev/sdc1: UUID="9cd380e5-daf3-4294-9b11-5d3d938b5345" TYPE="ext4" PARTLABEL="p1" PARTUUID="bb8a8b3e-5d44-4c4f-bf89-055efefeae
f7"
[root@localhost ~]# vi  /etc/fstab

#
# /etc/fstab
# Created by anaconda on Fri Aug 30 20:19:56 2024
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=97e92bf8-6b30-4260-9418-31a22fc40b49 /boot                   xfs     defaults        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
# UUID=49f30a3b-6d6a-4626-8cc3-84a2743a3453     /part1  ext4    defaults        0 0
UUID=9cd380e5-daf3-4294-9b11-5d3d938b5345       /part2  ext4    defaults        0 0
~
~
~
~
~
~
~
~
~
~
:wq!
[root@localhost ~]# mount -a
```

1. Used `**blkid /dev/sdc1**` to check the UUID.

2. Edited `**/etc/fstab**` with `**vi /etc/fstab**` to update it using the UUID.

3. Saved the changes and exited `**vi**` with `**:wq!**`.

4. Used `**mount -a**` to verify that all mounts were processed correctly.

5. This ensures that the file system is mounted automatically after the system reboots.

# Step 17

```
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
sdc               8:32   0    5G  0 disk
└─sdc1            8:33   0  2.1G  0 part /part2
sr0              11:0    1 1024M  0 rom
[root@localhost ~]# umount  /dev/sdc1
[root@localhost ~]# vi /etc/fstab
```

- 1. To delete a partition, unmount it first with **`umount /dev/sdc1`.**
- 2. Edit `/etc/fstab` with **`vi /etc/fstab`** to remove the entry for the partition.
- 3. Save the changes and exit `vi` with **`:wq!`.**

# Step 18

```
# /etc/fstab
# Created by anaconda on Fri Aug 30 20:19:56 2024
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=97e92bf8-6b30-4260-9418-31a22fc40b49 /boot                   xfs     defaults        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
# UUID=49f30a3b-6d6a-4626-8cc3-84a2743a3453     /part1  ext4    defaults        0 0
# UUID=9cd380e5-daf3-4294-9b11-5d3d938b5345     /part2  ext4    defaults        0 0
~
~
:wq!
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
sdc               8:32   0    5G  0 disk
└─sdc1            8:33   0  2.1G  0 part
```

1. Edited `/etc/fstab` with `vi /etc/fstab` and added `#` to comment out the UUID entry.

2. Used **`:wq`** to save the changes and exit `vi`.

3. Used **`lsblk`** to check if the partition was unmounted successfully.

4. Verified that the partition was successfully unmounted.

# Step 19

```
[root@localhost ~]# parted /dev/sdc1
GNU Parted 3.1
Using /dev/sdc1
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) p
Model: Unknown (unknown)
Disk /dev/sdc1: 2221MB
Sector size (logical/physical): 512B/512B
Partition Table: loop
Disk Flags:

Number  Start  End     Size    File system  Flags
 1      0.00B  2221MB  2221MB  ext4

(parted) rm 1
(parted) p
Model: Unknown (unknown)
Disk /dev/sdc1: 2221MB
Sector size (logical/physical): 512B/512B
Partition Table: loop
Disk Flags:

Number  Start  End  Size  File system  Flags
```

- 1. Used `**parted /dev/sdc**` to remove the partition.
- 2. Executed the command `**rm 1**` to remove partition 1 (where `**1**` specifies the partition number).
- 3. Used `**p**` in `parted` to check the partition table.
- 4. Verified that the partition was successfully removed.

# Step 20

```
Number  Start  End  Size  File system  Name  Flags

(parted) w
  align-check TYPE N                        check partition N for TYPE(min|opt) alignment
  help [COMMAND]                            print general help, or help on COMMAND
  mklabel,mktable LABEL-TYPE                create a new disklabel (partition table)
  mkpart PART-TYPE [FS-TYPE] START END      make a partition
  name NUMBER NAME                          name partition NUMBER as NAME
  print [devices|free|list,all|NUMBER]      display the partition table, available devices, free space, all found partitions,
        or a particular partition
  quit                                      exit program
  rescue START END                          rescue a lost partition near START and END

  resizepart NUMBER END                     resize partition NUMBER
  rm NUMBER                                 delete partition NUMBER
  select DEVICE                             choose the device to edit
  disk_set FLAG STATE                       change the FLAG on selected device
  disk_toggle [FLAG]                        toggle the state of FLAG on selected device
  set NUMBER FLAG STATE                     change the FLAG on partition NUMBER
  toggle [NUMBER [FLAG]]                    toggle the state of FLAG on partition NUMBER
  unit UNIT                                 set the default unit to UNIT
  version                                   display the version number and copyright information of GNU Parted
(parted) q
Information: You may need to update /etc/fstab.
```



```
[root@localhost ~]# lsblk
NAME                MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                   8:0    0    20G  0 disk
├─sda1                8:1    0     1G  0 part /boot
└─sda2                8:2    0    19G  0 part
  ├─centos-root     253:0    0    17G  0 lvm  /
  └─centos-swap     253:1    0     2G  0 lvm  [SWAP]
sdb                   8:16   0    10G  0 disk
└─sdb1                8:17   0    10G  0 part
sdc                   8:32   0     5G  0 disk
```

- 1. Used `**w**` in `parted` to confirm the partition removal and write changes to the partition table.
- 2. Used `**lsblk**` to verify that the partition was successfully removed.



# Project scope

1. Explain the mkfs command and create the file system.
2. Mount the file system and unmount the file system.
3. Understand the concepts of LVM.
4. Create PV, VG, LV and manage the file system.

# Explain the mkfs command and create the file system

**What is ‘mkfs’ :** The ‘mkfs’ command is used to create a file system on a disk or partition , formatting it to be used for storing data.

**Prerequisites for Using ‘mkfs’:** The disk or partition must not be mounted ; it should be an unmounted empty disk or partition to perform this command.

**Verify File System Creation :** Use the ‘lsblk –f ‘ command to check and confirm the file system has been created successfully.

# Step 1

```
[root@localhost ~]# lsblk -f
NAME            FSTYPE      LABEL UUID                                   MOUNTPOINT
sda
├─sda1          xfs               97e92bf8-6b30-4260-9418-31a22fc40b49   /boot
└─sda2          LVM2_member       BSJqvd-RjIz-oSjb-otcJ-3Ebh-Uj2G-EBWa7O
  ├─centos-root xfs               7f20de15-9d76-4f74-8f46-5f1a3abd486f   /
  └─centos-swap swap              4f46c356-6dc8-405d-932b-879cc767cadc   [SWAP]
sdb
└─sdb1
sdc
sr0
```

```
[root@localhost ~]# mkfs.ext4 /dev/sdb1
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621184 blocks
131059 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2151677952
80 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

**1. Verify Disk Status:**

- I used `**lsblk -f**` to check if the disk is mounted. The disk `/dev/sdb1` is not mounted.

**2. Select the Disk:**

- Since `**/dev/sdb1**` is unmounted, I will use this disk to perform the `**mkfs**` command.

**3. Create File System:**

- I used the command `**mkfs.ext4 /dev/sdb1**` to format the partition with the ext4 file system.

# Step 2

```
[root@localhost ~]# lsblk -f
NAME            FSTYPE      LABEL UUID                                   MOUNTPOINT
sda
├─sda1          xfs               97e92bf8-6b30-4260-9418-31a22fc40b49   /boot
└─sda2          LVM2_member       BSJqvd-RjIz-oSjb-otcJ-3Ebh-Uj2G-EBWa70
  ├─centos-root xfs               7f20de15-9d76-4f74-8f46-5f1a3abd486f   /
  └─centos-swap swap              4f46c356-6dc8-405d-932b-879cc767cadc   [SWAP]
sdb
└─sdb1          ext4              be6aa3b0-97fc-48a5-81a6-98c69f8ea1af
sdc
sr0
```

**1. Verify File System Creation:**

- I used `lsblk -f` to check if the ext4 file system was created successfully.

**2. Confirm Partition Status:**

- The partition `/dev/sdb1` is shown as successfully formatted with the ext4 file system.

# Mount the file system and unmount the file system.

**What is ‘mkfs’ :** The ‘mkfs’ command is used to create a file system on a disk or partition , formatting it to be used for storing data.

**Prerequisites for Using ‘mkfs’:** The disk or partition must not be mounted ; it should be an unmounted empty disk or partition to perform this command.

**Verify File System Creation :** Use the ‘lsblk –f ‘ command to check and confirm the file system has been created successfully.

# Step 1

```
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
```

```
[root@localhost ~]# mkfs.ext4  /dev/sdb1
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621184 blocks
131059 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2151677952
80 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
```

I used `**lsblk**` to confirm that the file system was not mounted, then performed `**mkfs.ext4 /dev/sdb1**` to format the partition with the EXT4 file system.

# Step 2

```
[root@localhost ~]# mkdir  /part1
[root@localhost ~]# mount /dev/sdb1/      /part1
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part /part1
sr0              11:0    1 1024M  0 rom
[root@localhost ~]# udevadm settle
[root@localhost ~]# partprobe /dev/sdb
[root@localhost ~]# blkid  /dev/sdb1
/dev/sdb1: UUID="49f30a3b-6d6a-4626-8cc3-84a2743a3453" TYPE="ext4"
[root@localhost ~]# vi /etc/fstab
```

1. Used `**mkdir /part1**` to create a directory.

2. Mounted the disk on `**/part1**` with `**mount /dev/sdb1 /part1**`.

3. Ran `**udevadm settle**` and `**partprobe /dev/sdb**` to update the system's partition table.

4. Checked the UUID with `**blkid /dev/sdb1**`.

5. Updated `**/etc/fstab**` **using** `**vi**` to ensure the file system mounts automatically after a reboot.

# Step 3

```
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=97e92bf8-6b30-4260-9418-31a22fc40b49 /boot                   xfs     defaul
ts        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
UUID=49f30a3b-6d6a-4626-8cc3-84a2743a3453       /part1  ext4    defaults
0 0
~
~
:wq!
```

```
[root@localhost ~]# vi /etc/fstab
[root@localhost ~]# mount -a
```

```
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part /part1
sr0              11:0    1 1024M  0 rom
```

1. Used `**mount -a**` to confirm that the mount was successful.

2. Ran `**lsblk**` to check if the partition was mounted correctly.

3. Verified that the partition was successfully mounted on `***/part1***`.

# Step 4

```
[root@localhost ~]# umount /dev/sdb1
[root@localhost ~]# vi /etc/fstab
# /etc/fstab
# Created by anaconda on Fri Aug 30 20:19:56 2024
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=97e92bf8-6b30-4260-9418-31a22fc40b49 /boot                   xfs     defaul
ts        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
# UUID=49f30a3b-6d6a-4626-8cc3-84a2743a3453     /part1  ext4    defaults
0 0
~
~
~
~
~
~
~
~
~
:wq!
```

1. Used `**umount /dev/sdb1**` to unmount the partition.

2. Edited `**/etc/fstab**` with `**vi /etc/fstab**` to update the file system table.

3. Saved the changes and exited `**vi**`.

# Step 5

```
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
sr0              11:0    1 1024M  0 rom
[root@localhost ~]#
```

1. Used `**lsblk**` to check and verify that the partition was unmounted.

2. Confirmed that the partition was successfully unmounted.

# Understand the concepts of LVM.

**What is LVM :** LVM(logical Volume Management) is a system for managing disk storage that allows for flexible allocation and management of disk space by abstracting physical storage into logical volumes.

**Key Components of LVM**

**Benefits of LVM**

Basic Commands

Workflow of Creating and Removing LVM

# Key Components of LVM

**Physical Volume (PV):** The basic storage unit in LVM, typically a disk or partition that is initialized for use by LVM.

**Volume Group (VG):** A collection of Physical Volumes that creates a pool of storage space from which Logical Volumes can be allocated.

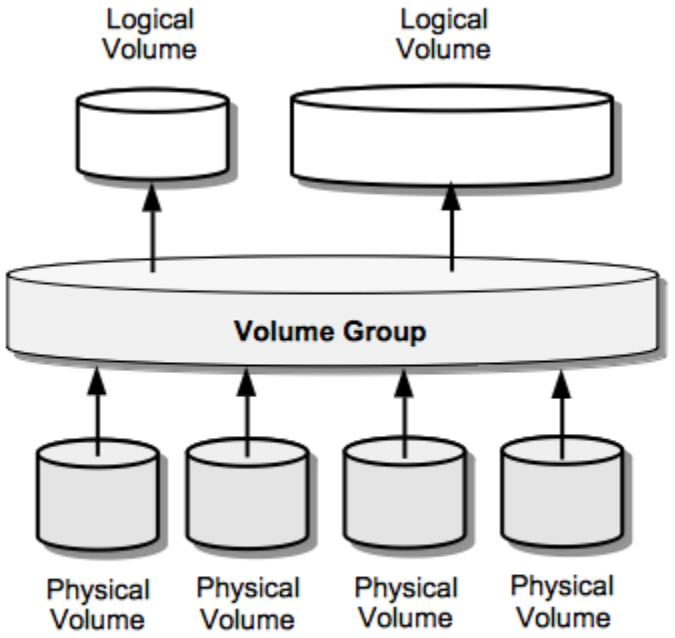
**Logical Volume (LV):** The logical partitions created from the Volume Group, which are used like traditional partitions but with flexible size adjustments.

# Benefits of LVM

**Flexibility:** Easily resize logical volumes without the need to repartition the disk.

**Snapshots:** Create snapshots of volumes to back up or test data without affecting the original volume.

**Dynamic Storage Management:** Combine multiple disks into a single storage pool and manage it more efficiently




Logical Volume Management Components

# Basic Commands

1. **Physical Volume (PV) Commands:** Use `**pvcreate**` to initialize a disk as a Physical Volume**.** `**pvdisplay**` shows details about Physical Volumes, while `**pvremove**` removes LVM configuration from a disk. The `**pvs**` command provides a summary of all Physical Volumes.

**2. Volume Group (VG) Commands:** Create a Volume Group with `**vgcreate**`. View its details using `**vgdisplay**` and delete it with `**vgremove**` after removing all Logical Volumes. The `**vgs**` command offers a summary of Volume Groups.

**3. Logical Volume (LV) Commands:**Create Logical Volumes with `**lvcreate**`, view details with `**lvdisplay**`**,** and remove them using `**lvremove**`**.** Resize with `**lvextend**` or `**lvreduce**`**,** and use `**lvs**` for a summary of all Logical Volumes.

# Workflow of Creating and Removing LVM

**Creating LVM**

**Physical Volume(PV)**

**Volume Group(VG)**

**Logical Volume(LV)**

**Removing LVM**

**Logical Volume(LV)**

**Volume Group(VG)**

**Physical Volume(PV)**

# Create PV, VG, LV and manage the file system

**Prepare an Empty Disk :** Ensure the disk you will use is empty and not mounted. Verify the disk status using the `lsblk` command to ensure it is available and unmounted.

**Manage and Format the disk:** I used the `fdisk` command to partition the disk in preparation for creating LVM.

**Create LVM Components:** Create a Physical Volume (PV), a Volume Group (VG), and a Logical Volume (LV), then format and mount the LV.

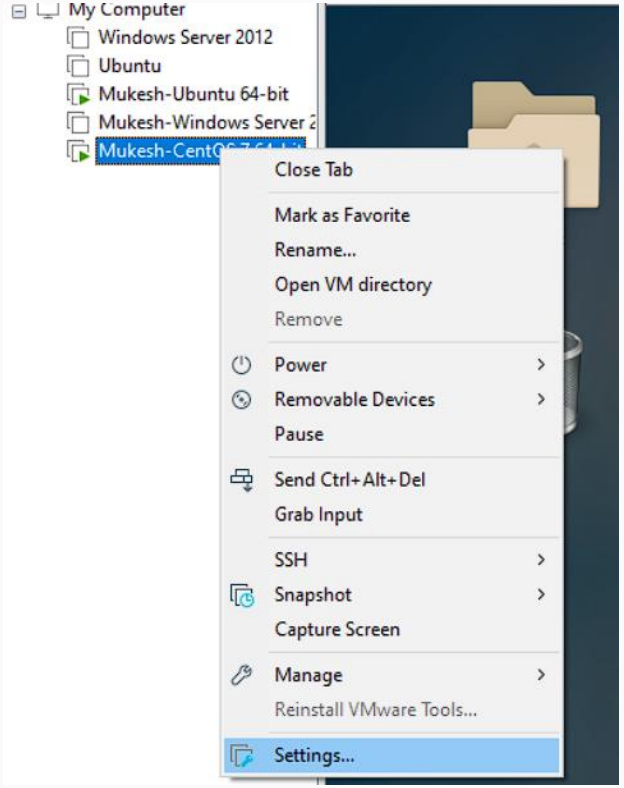
**Delete LVM Components :** Remove the Logical Volume (LV), Volume Group (VG), and Physical Volume (PV), and then delete the associated partitions.

# Prepare an Empty Disk

```
File  Edit  View  Search  Terminal  Help
[mukesh@localhost ~]$ lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sr0              11:0    1 1024M  0 rom
[mukesh@localhost ~]$
```

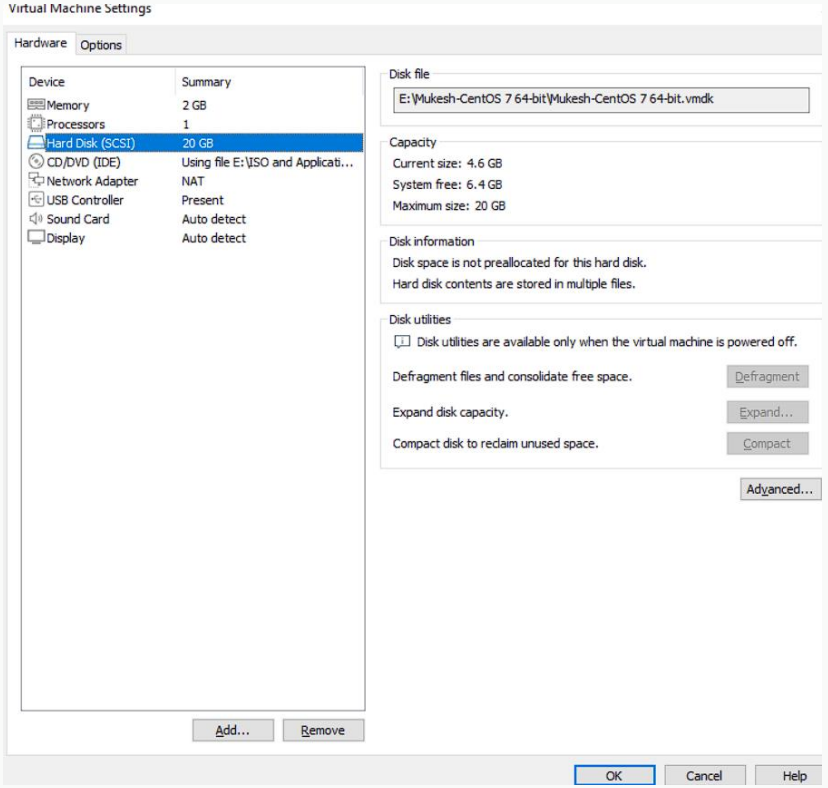
I used `**lsblk**` to check if the disk is empty. If it is not, we will manually add a virtual disk.

# Step 1



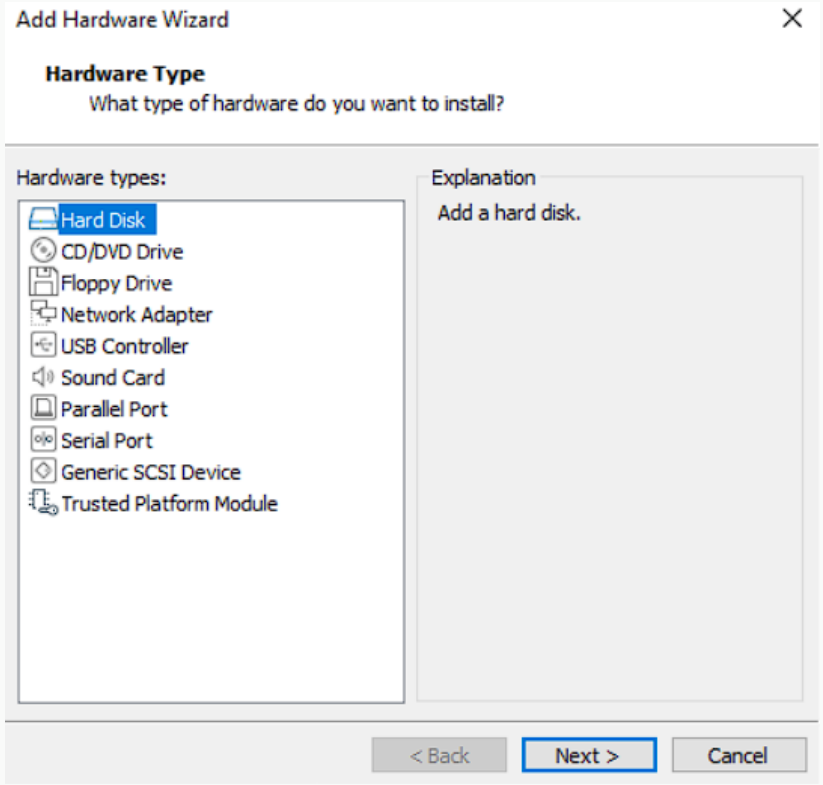

To add a disk manually, right-click on the OS machine name, select **"Settings,"** and proceed with the next steps.

# Step 2



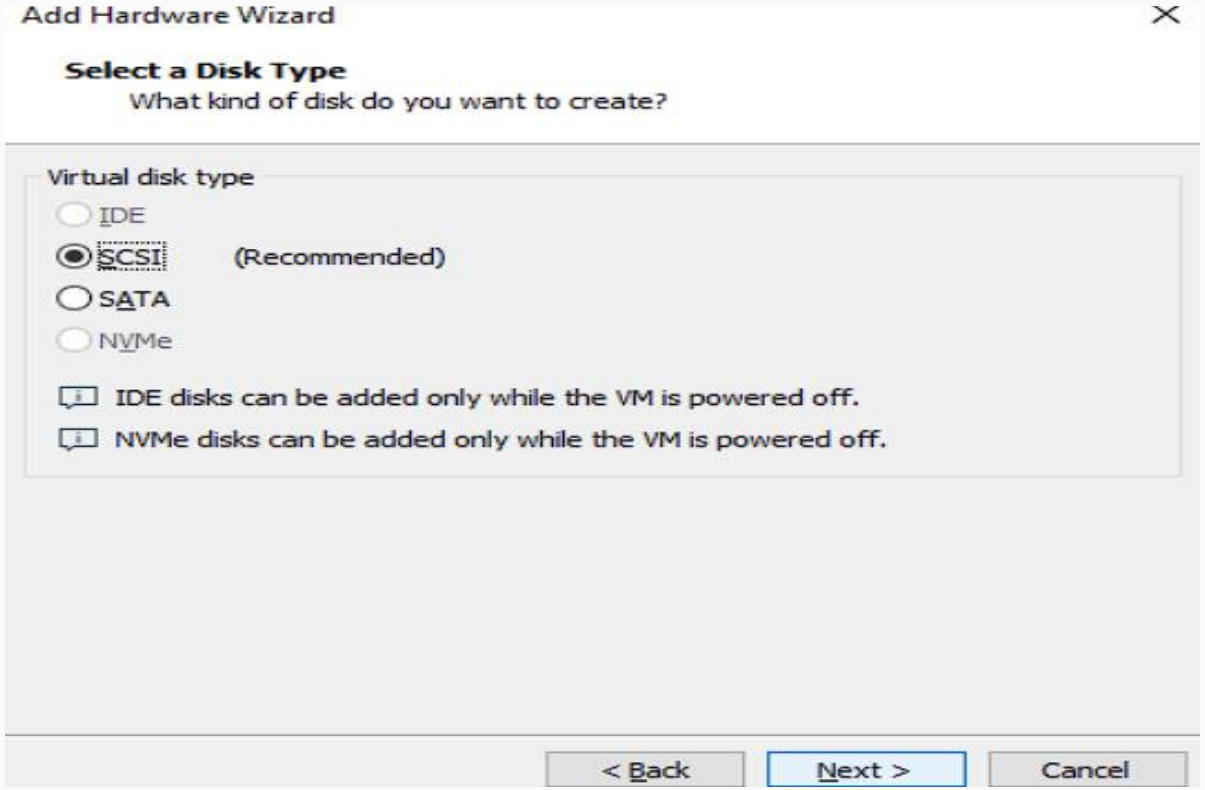

Then, select the **"Hard Disk"** option and click on "Add" and proceed with the next steps.

# Step 3



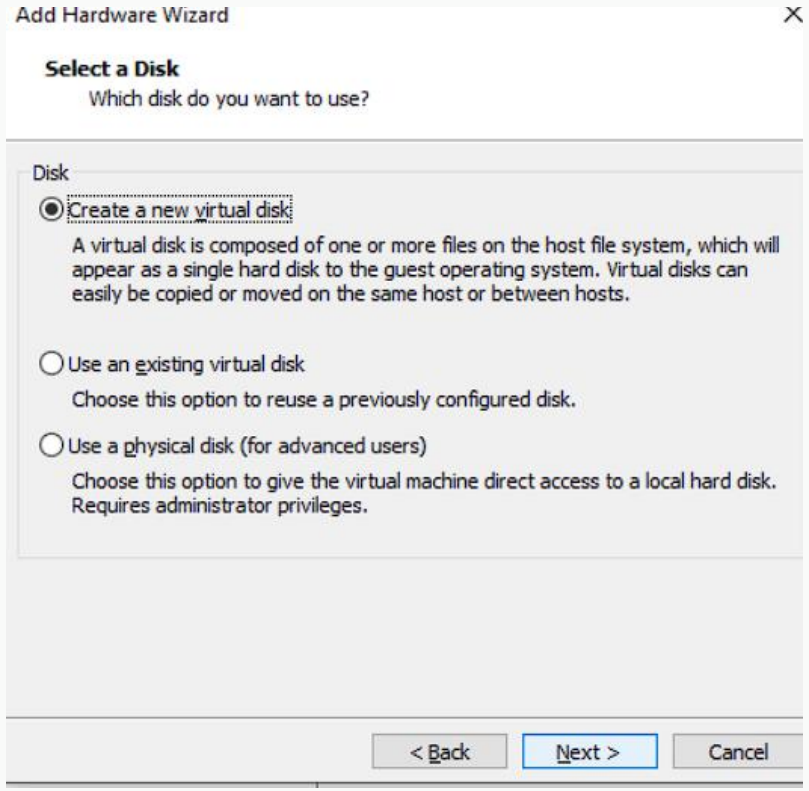

Again, select **"Hard Disk"** and click on **"Next,"** then wait for the subsequent steps.

# Step 4



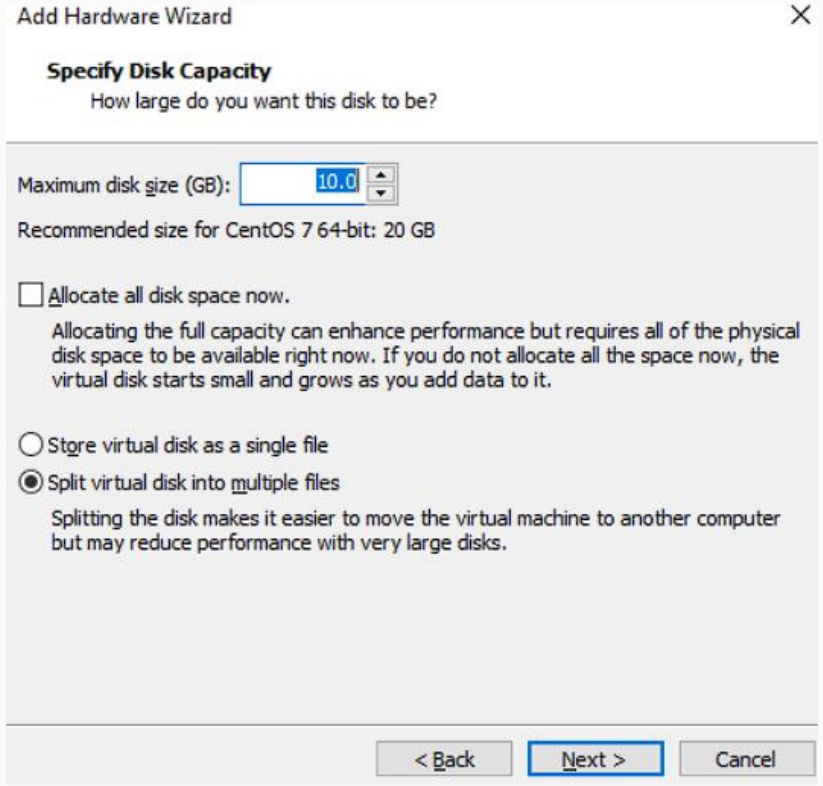

Then, select "SCSI" (which is already recommended), and click on "Next."

# Step 5



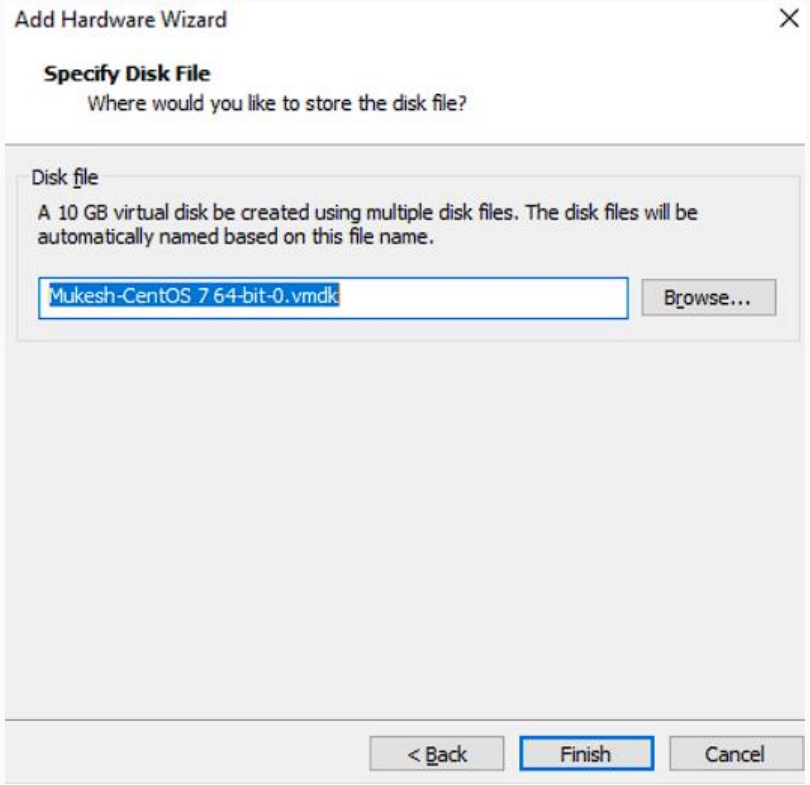

Select **"Create a new virtual disk,"** and then click on "Next."

# Step 6



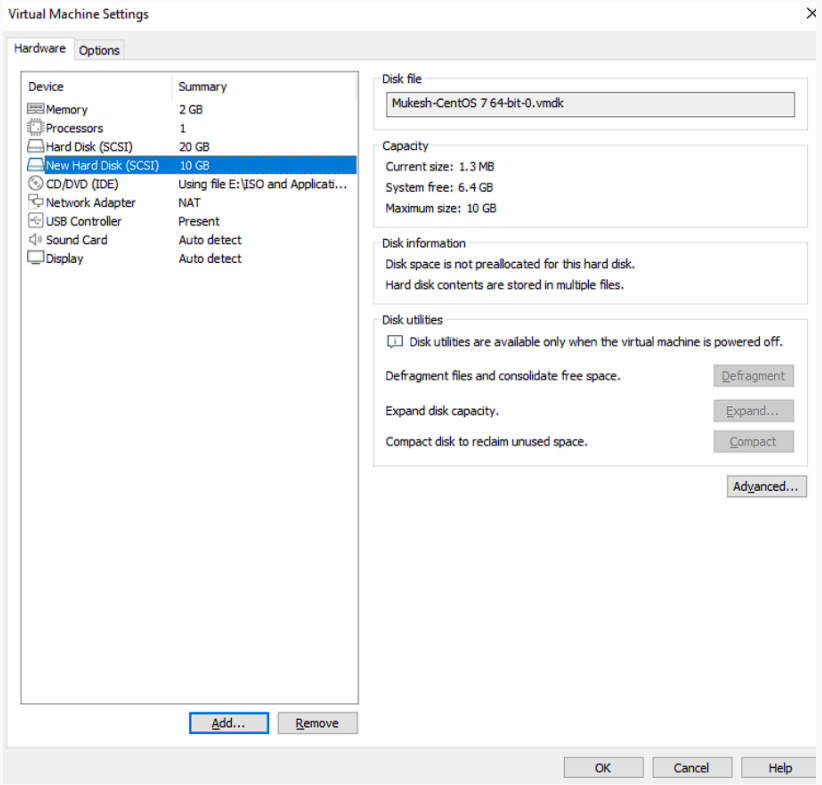

Select **"Split virtual disk into multiple files,"** set the **volume size**, and click **"Next."**

# Step 7



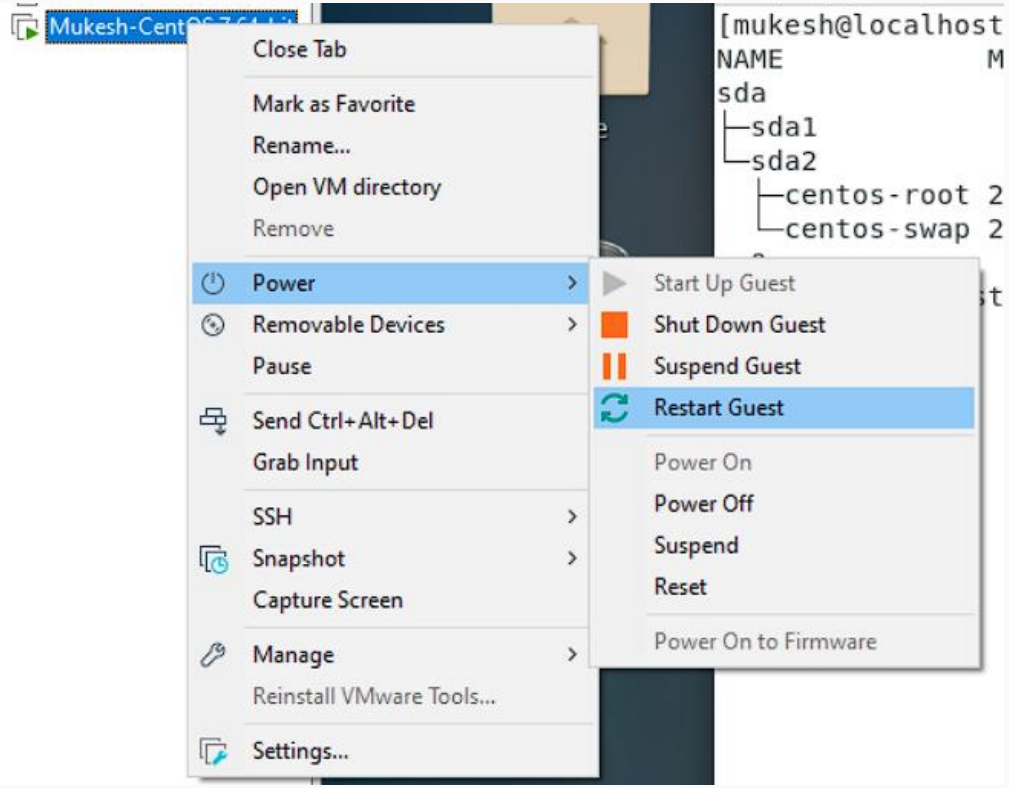

Confirm the file name, click **"Finish,"** and proceed to the next step.

# Step 8




After confirming all the details, click **“OK."**

# Step 9




To apply the changes, **right-click** on the OS machine name, click on **"Power,"** select **"Restart,"** and then reboot the system.

# Manage and Format the disk

```
File  Edit  View  Search  Terminal  Help
[mukesh@localhost ~]$ lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
sr0              11:0    1 1024M  0 rom
[mukesh@localhost ~]$
```

Now, use the **`lsblk`** command to verify if the disk has been created.

# Step 1

```
[root@localhost ~]# fdisk /dev/sdb
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table
Building a new DOS disklabel with disk identifier 0xd1e6bbdf.

Command (m for help): p

Disk /dev/sdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0xd1e6bbdf

   Device Boot      Start         End      Blocks   Id  System

Command (m for help):
```

Use `**fdisk /dev/sdb**` to manage partitions, then type `**p**` to print the partition table.

# Step 2

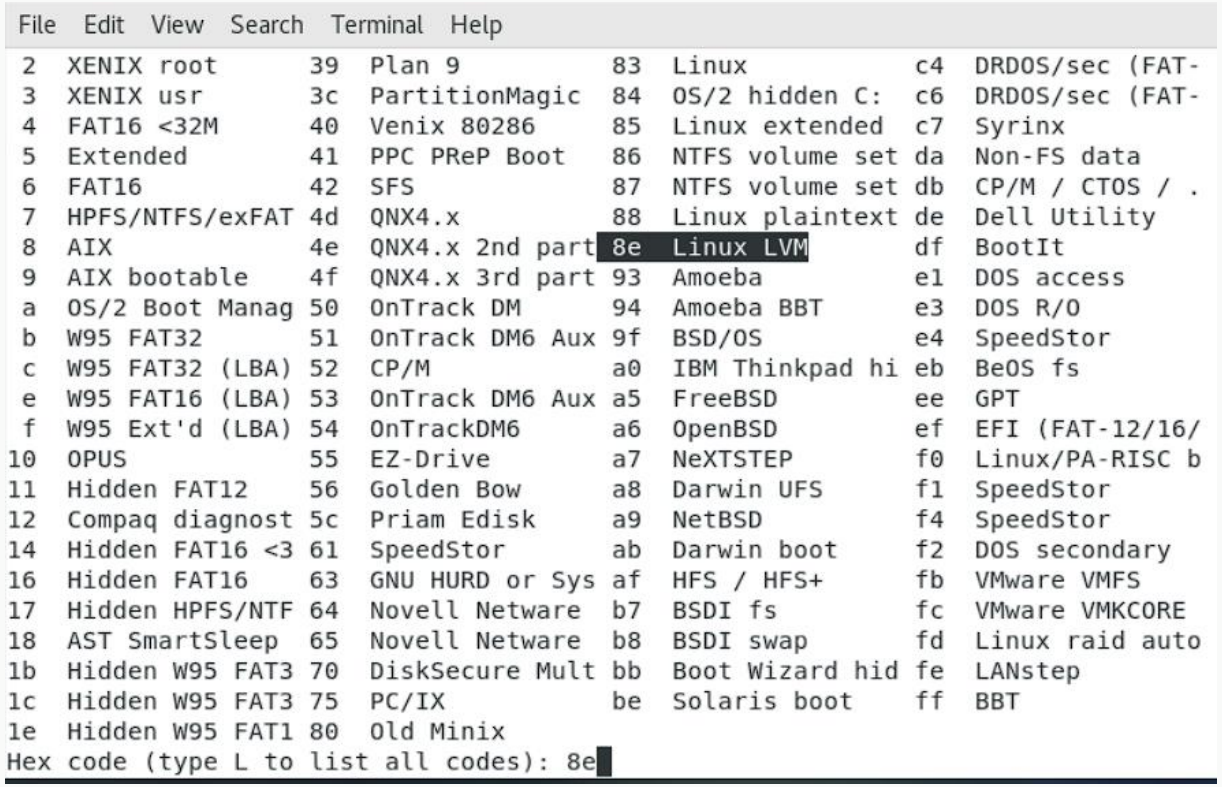
```
Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-20971519, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519):
Using default value 20971519
Partition 1 of type Linux and of size 10 GiB is set
Command (m for help): p

Disk /dev/sdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0xd1e6bbdf

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048    20971519    10484736   83  Linux

Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): L
```

**1. Create a New Partition:** Use `**n**` to start creating a new partition, then `**p**` to select a primary partition type.

**2. Set Partition Number and Sector:** Accept default values for the partition number, first sector, and last sector, then use `**p**` to print the updated partition table.

**3. Change and List Partition Type:** Use `**t**` to change the partition type and `**L**` to list available partition types.

# Step 3




Here, select `**8e**` as the partition type, which corresponds to the Linux LVM type.

# Step 4

```
Hex code (type L to list all codes): 8e
Changed type of partition 'Linux' to 'Linux LVM'

Command (m for help): p

Disk /dev/sdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0xd1e6bbdf

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048    20971519    10484736   8e  Linux LVM

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

Now, use `**p**` to print the partition table and verify that the partition has been created. Then, use `**w**` to write the changes to the disk and exit.

# Create LVM Components

```
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0   19G  0 part
  ├─centos-root 253:0    0   17G  0 lvm  /
  └─centos-swap 253:1    0    2G  0 lvm  [SWAP]
sdb               8:16   0   10G  0 disk
└─sdb1            8:17   0   10G  0 part
sr0              11:0    1 1024M  0 rom
```

Now, use `**lsblk**` to check that `**/dev/sdb1**` has been created. Once confirmed, proceed to the next step.

# Step 1

```
[root@localhost ~]# pvs
  PV         VG     Fmt  Attr PSize   PFree
  /dev/sda2  centos lvm2 a--  <19.00g    0
[root@localhost ~]# pvcreate /dev/sdb1
  Physical volume "/dev/sdb1" successfully created.
[root@localhost ~]# pvs
  PV         VG     Fmt  Attr PSize   PFree
  /dev/sda2  centos lvm2 a--  <19.00g      0
  /dev/sdb1         lvm2 ---  <10.00g <10.00g
```

```
[root@localhost ~]# vgs
  VG     #PV #LV #SN Attr   VSize   VFree
  centos   1   2   0 wz--n- <19.00g    0
[root@localhost ~]# vgcreate vg0 /dev/sdb1
  Volume group "vg0" successfully created
[root@localhost ~]# vgs
  VG     #PV #LV #SN Attr   VSize   VFree
  centos   1   2   0 wz--n- <19.00g      0
  vg0      1   0   0 wz--n- <10.00g <10.00g
```

**1. Physical Volumes (PV):** Use `**pvs**` to display existing PVs, then initialize `/dev/sdb1` as a PV with `**pvcreate /dev/sdb1**`.

**2. Volume Groups (VG):** Use `**vgs**` to display existing VGs, then create a VG named `vg0` with `**vgcreate vg0 /dev/sdb1**`.

# Step 2

```
[root@localhost ~]# lvs
  LV   VG     Attr       LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync
vert
  root centos -wi-ao---- <17.00g

  swap centos -wi-ao----   2.00g

[root@localhost ~]# lvcreate -L 500M -n lvo vg0
  Logical volume "lvo" created.
[root@localhost ~]# lvs
  LV   VG     Attr       LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync
vert
  root centos -wi-ao---- <17.00g

  swap centos -wi-ao----   2.00g

  lvo  vg0    -wi-a----- 500.00m
```

1. List Logical Volumes (LV): Use `**lvs**` to display existing Logical Volumes.

2. Create Logical Volume (LV): Run `**lvcreate -L 500M -n lv0 vg0**` to create a Logical Volume named `**lv0**` with a size of 500MB in the Volume Group `**vg0**`.

# Step 3

```
[root@localhost ~]# pvs
  PV         VG     Fmt  Attr PSize   PFree
  /dev/sda2  centos lvm2 a--  <19.00g     0
  /dev/sdb1  vg0    lvm2 a--  <10.00g <9.51g
[root@localhost ~]# vgs
  VG     #PV #LV #SN Attr   VSize   VFree
  centos   1   2   0 wz--n- <19.00g     0
  vg0      1   1   0 wz--n- <10.00g <9.51g
[root@localhost ~]# lvs
  LV   VG     Attr       LSize   Pool Origin Data%
vert
  root centos -wi-ao---- <17.00g

  swap centos -wi-ao----   2.00g

  lvo  vg0    -wi-a----- 500.00m
```

**1. Confirm Physical Volumes (PV):** Use `pvs` to verify the Physical Volumes.

**2. Confirm Volume Groups (VG):** Use `vgs` to check the Volume Groups.

**3. Confirm Logical Volumes (LV):** Use `lvs` to verify the Logical Volumes.

# Delete LVM Components

```
[root@localhost ~]# lvremove /dev/vg0/lvo
Do you really want to remove active logical volume vg0/lvo? [y/n]: y
  Logical volume "lvo" successfully removed
[root@localhost ~]# vgremove vg0
  Volume group "vg0" successfully removed
[root@localhost ~]# pvremove   /dev/sdb1
  Labels on physical volume "/dev/sdb1" successfully wiped.
```

```
[root@localhost ~]# pvs
  PV         VG     Fmt  Attr PSize   PFree
  /dev/sda2  centos lvm2 a--  <19.00g    0
[root@localhost ~]# vgs
  VG     #PV #LV #SN Attr   VSize   VFree
  centos   1   2   0 wz--n- <19.00g    0
[root@localhost ~]# lvs
  LV   VG     Attr       LSize   Pool Origin Data%
vert
  root centos -wi-ao---- <17.00g

  swap centos -wi-ao----   2.00g
```

**1. Delete LVM Components:** Use `**lvremove /dev/vg0/lv0**` to remove the Logical Volume, `**vgremove vg0**` to delete the Volume Group, and `**pvremove /dev/sdb1**` to remove the Physical Volume.

**2. Verify Deletion:** Use `**pvs**`, `**vgs**`, and `**lvs**` to confirm that the Physical Volume, Volume Group, and Logical Volume have been successfully removed.

# Summary and Conclusion

- Summary
- Prerequisites & Project Requirements
- Deliverables
- Conclusion

# Summary

This project focuses on understanding and managing the File System in Linux, including partitioning, file system creation, and Logical Volume Management (LVM). The following key aspects were covered:

- **Understanding File System Architecture:** We explored the structure and components of file systems in Linux, including how data is organized and accessed.
- **Partitioning and File System Management:** Detailed instructions were provided on creating, mounting, and managing partitions. The `parted` command was utilized for partition management, and `mkfs` for file system creation.
- **Logical Volume Management (LVM):** The project covered the principles of LVM, including the creation and management of Physical Volumes (PV), Volume Groups (VG), and Logical Volumes (LV). Practical exercises demonstrated how to set up and manage LVM effectively.
- **Disk Space Management:** Strategies for managing and optimizing disk space were discussed, ensuring efficient use of available resources.

# Prerequisites & Project Requirements

To successfully complete this project, the following were necessary:

- **Knowledge of Volume Manager:**

  - Understand how Logical Volume Management (LVM) works to create and manage disk volumes flexibly.

- **Basic Understanding of File System and Types:**

  - Know what file systems are (e.g., ext4, XFS) and their roles in organizing and storing data on disks.

- **Understanding of User Commands and Administrative Commands:**

  - Be familiar with basic commands for navigating and managing files, as well as advanced commands for system administration tasks.

# Deliverables

The project produced the following deliverables:

- **Create and Delete File Partitions:**

  - Know how to divide a disk into separate sections and how to remove existing disk partitions.

- **Create and Mount File System:**

  - Set up a file system on a partition and ensure it is accessible by mounting it.

- **Manage Disk Space:**

  - Efficiently handle and optimize available disk space in the file system.

- **Create and Delete LVM Components:**

  - Set up and remove Logical Volume Management (LVM) elements, including Physical Volumes (PV), Volume Groups (VG), and Logical Volumes (LV).

# Conclusion

The project provided a thorough understanding of Linux file systems and partitioning. By exploring the architecture, management practices, and LVM setup, participants gained valuable skills in managing disk space and configuring systems efficiently. The practical exercises and comprehensive documentation ensure that participants are well-equipped to handle real-world file system management challenges in Linux environments.

This concluding section encapsulates the essence of what was covered in the project, highlighting key areas and summarizing the overall outcome. It offers a cohesive view of the project's requirements, deliverables, and objectives without needing to list each section separately.

# THANK YOU

Mukesh Yadav

Batch name : Data Centre - Batch 7
Cloud username : 24NAG2180_U07

Git username : b166user05