Devin Young

Bowie State University

CTEC 435

Professor Anthony

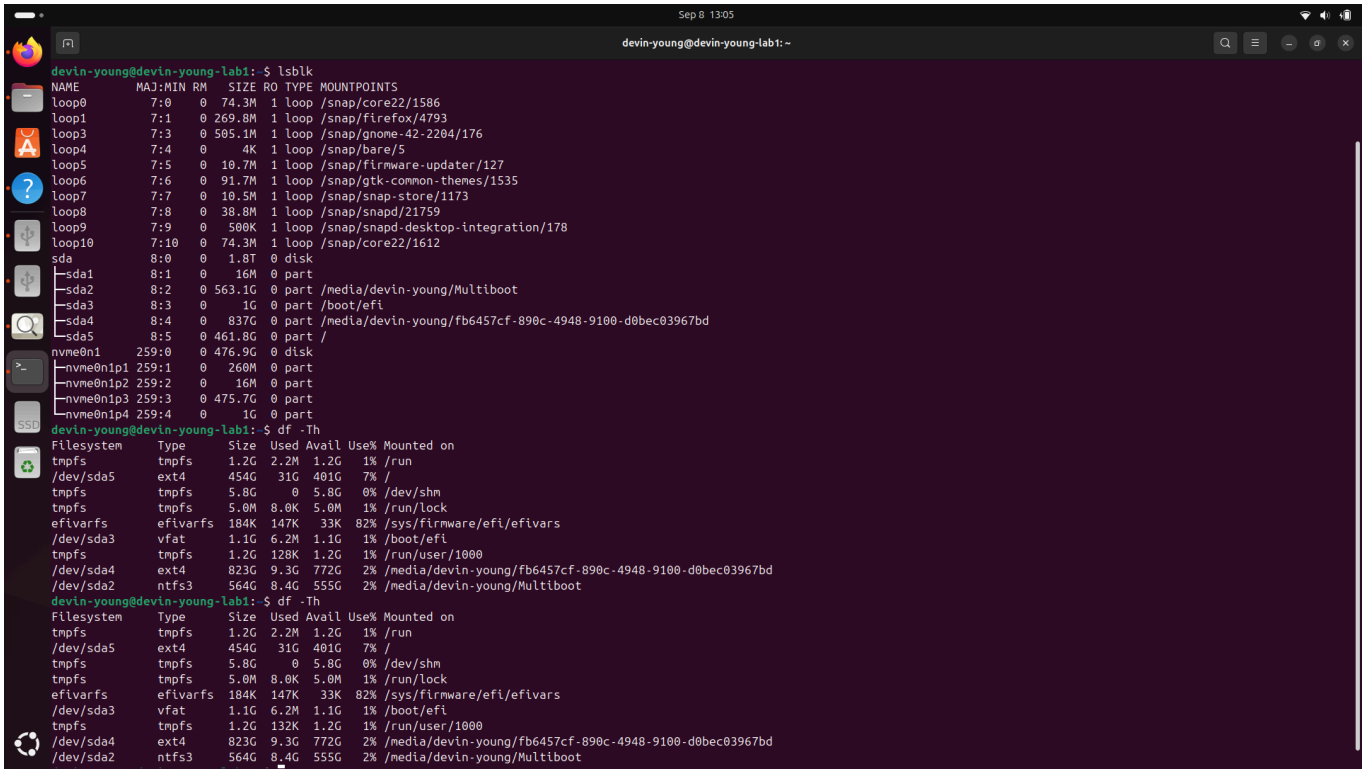
Part 1: Exploring Linux Filesystems

Task 1: Identify Filesystem Types

1. Use the `lsblk` and `df -Th` commands to list all mounted filesystems on your system.

2. Identify the filesystem types used for each partition (e.g., ext4, xfs, etc.).

3. Document the output and briefly describe the characteristics of each filesystem type identified.



Filesystem Types for Partitions

1. /dev/sda2 = ntfs3 (filesystem type)
2. /dev/sda3 = vfat (filesystem type)
3. /dev/sda4 = ext4 (filesystem type)

4. /dev/sda5 = ext4 (filesystem type)
5. tmpfs = tmpfs (filesystem type)
6. efivarfs = efivarfs (filesystem type)

Characteristics of FIlesystem Types

1. **ntfs3 (New Technology File System)**
    - Read-write driver for NTFS that works with Linux. Features include supporting all file operations (creating, altering, renaming, moving, and deleting files)
    - Supports compression and system-level encryption
    - Allows the alteration of access control lists and permissions
    - Supports extended attributes (such as system.ntfs_security and system.ntfs_attrib)
    - Supports native journal replaying
    - Supports NFS export of mounted NTFS volumes
    - Developed from scratch in 2020 to be a part of the Linux kernel (written in C language)
2. **vfat (Virtual FIle Allocation Table)**
    - Windows file system format used in storage devices
    - Supports file names up to 255 characters (which includes spaces)
    - Each directory entry represents a file or subdirectory
    - File attributes are stored in the directory entry (file name, size, etc.)
    - Available in three types (FAT12, FAT16, FAT 32)
3. **ext4 (Fourth Extended Filesystem)**
    - Can support volumes up to 1 exbibyte and files up to 16 tebibytes

- Uses extents to allocate file blocks (improves performance for large files)
- Backward compatible with ext3 and ext2 (can mount file systems created with those versions)
- Delays block allocation until data is flushed to disk
- Uses multiblock allocation (allocate many blocks in one call)
- Allows an unlimited number of subdirectories

4. **tmpfs (Temporary FIlesystem)**
    - All files are temporary and not saved to the hard drive
    - Stores data in volatile memory (such as RAM)
    - Directories mounted as tmpfs can speed up access to the files within those directories
    - Size parameter can be changed on remount
5. **efivarfs**
    - Linux kernel filesystem that allows users to modify, create, and delete UEFI variables
    - efivarfs files that are not well-known standardized variables are created as immutable files
    - No size limit
    - Supports UEFI secure boot
    - Introduced in kernel 3.8

Task 2: Mounting and Unmounting Filesystems

1. Create a new partition using a virtual disk or an existing unpartitioned space.

2. Format the new partition with a filesystem of your choice (e.g., ext4, xfs).

   - Use `mkfs.ext4` or `mkfs.xfs` commands for formatting.

3. Mount the new partition to a directory of your choice (e.g., `/mnt/newdisk`).

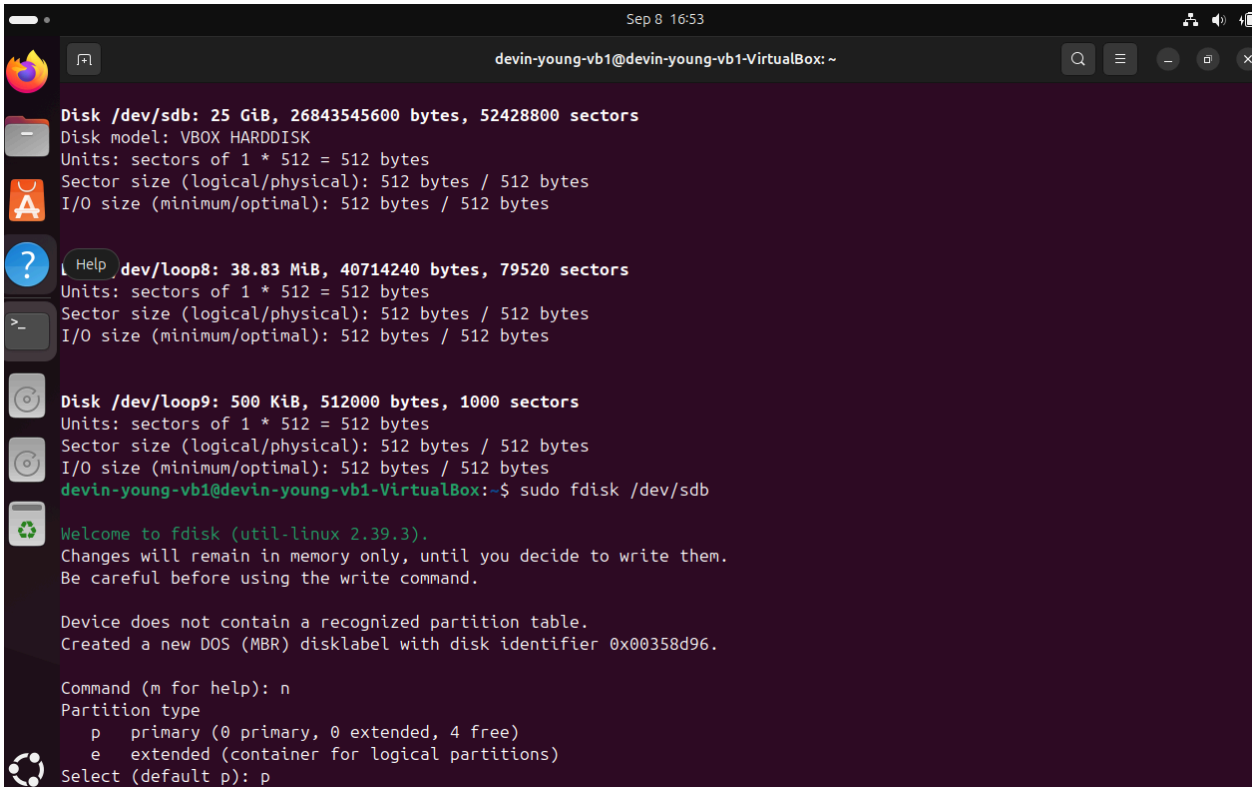
   - Use the `mount` command to mount the partition.

4. Verify that the partition is mounted using the `df -h` or `mount` command.

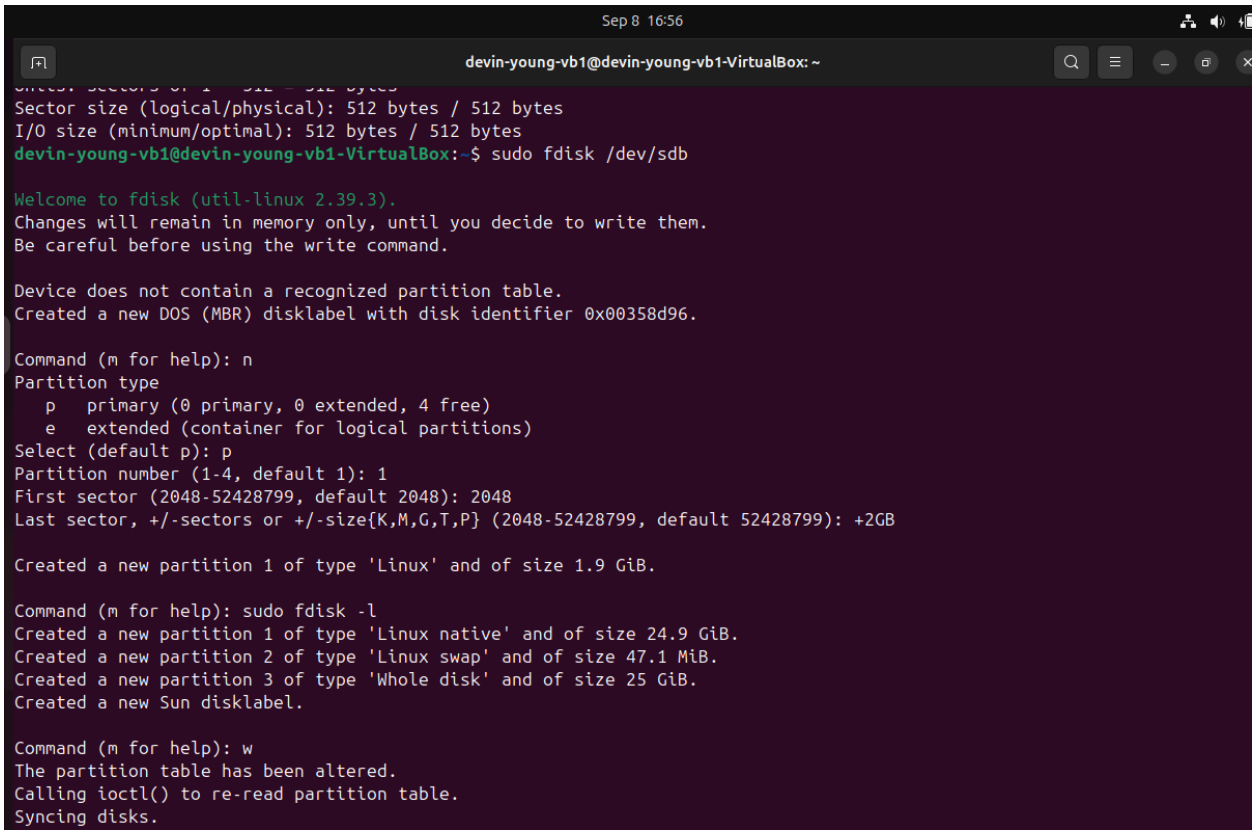
5. Unmount the partition using the `umount` command.

6. Document each step, including the commands used and the output.

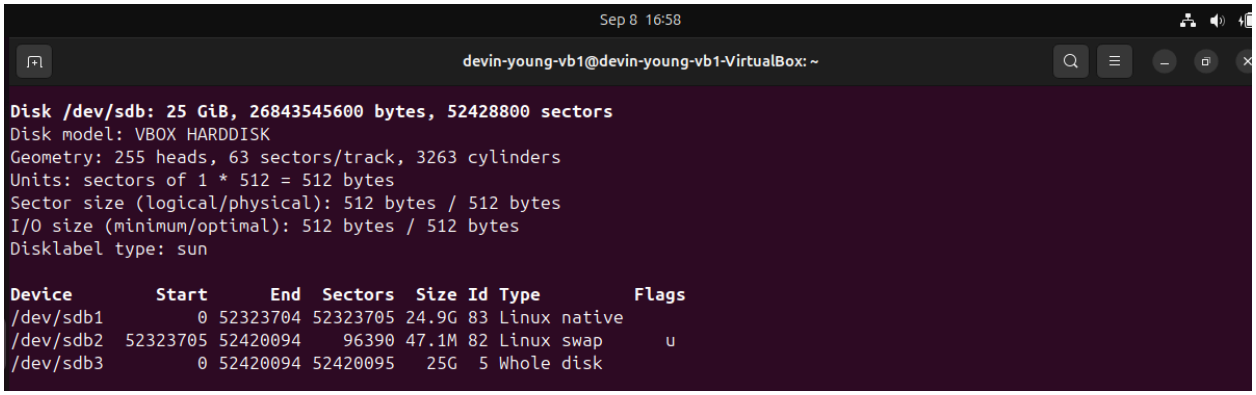
Step 1 - Created an additional virtual disk in virtual box on my already existing Ubuntu VM & Initiated the partition creation process with the sudo fdisk /dev/sdb command



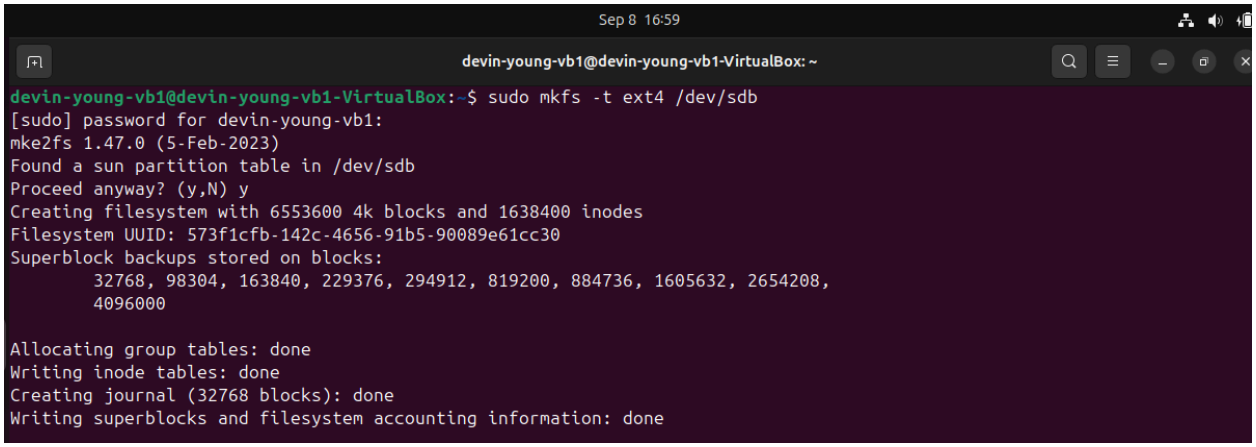
Step 2 - Completion of the partitioning of my virtual hard disk (sdb)



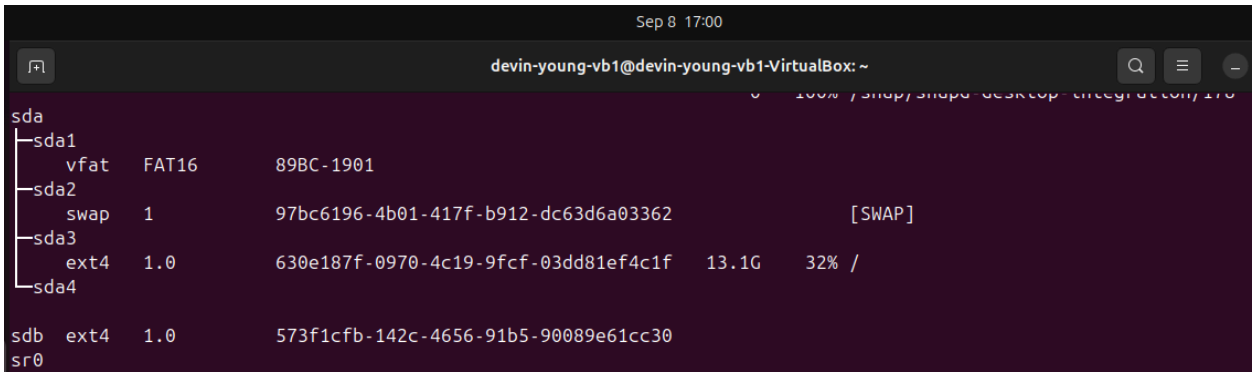
Step 3 - Confirmation that the virtual hard disk has been partitioned



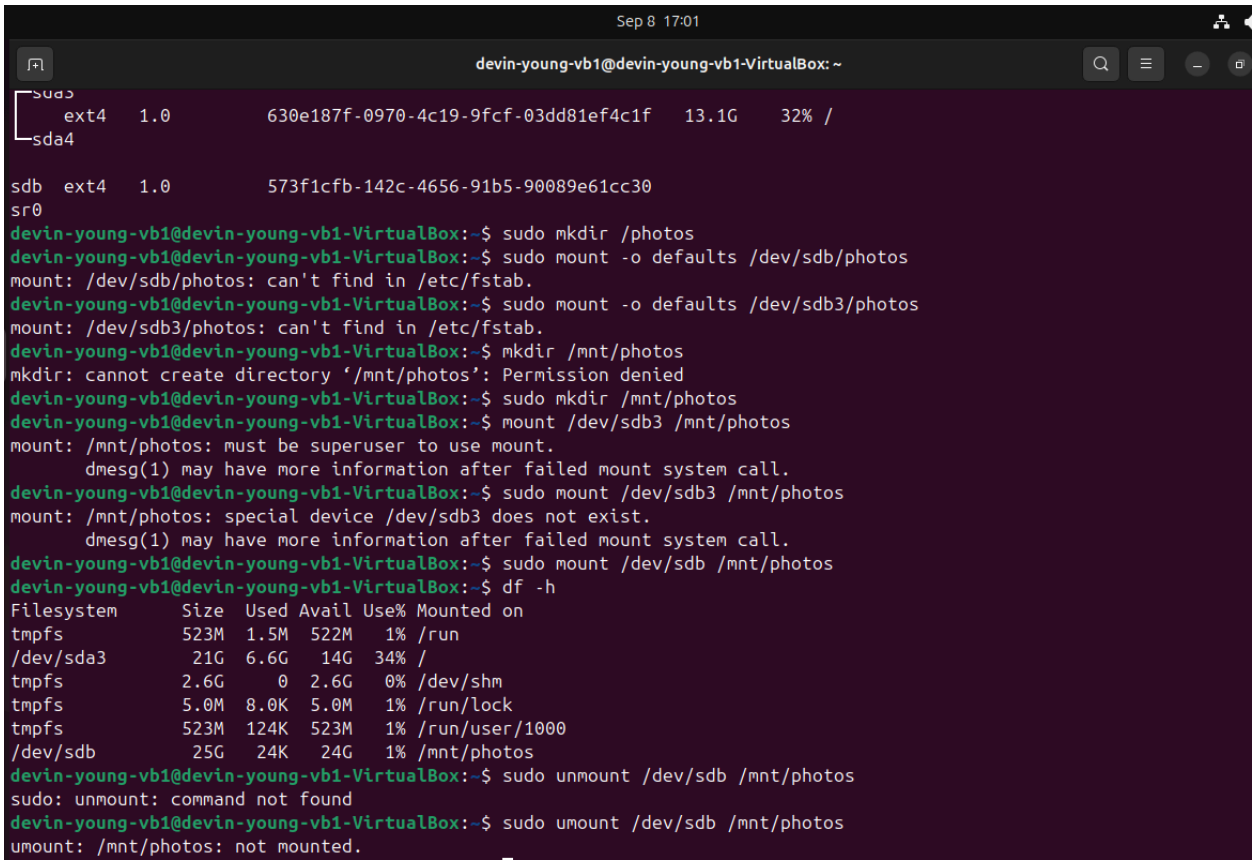
Step 4 - Formatted the partition with an ext4 filesystem



Step 5 - Confirmation that step 4 has been completed



Step 6 - Mounted the partition to a directory I created & unmounted it as directed



Part 2: Managing Linux Filesystems

Task 3: Disk Space Management

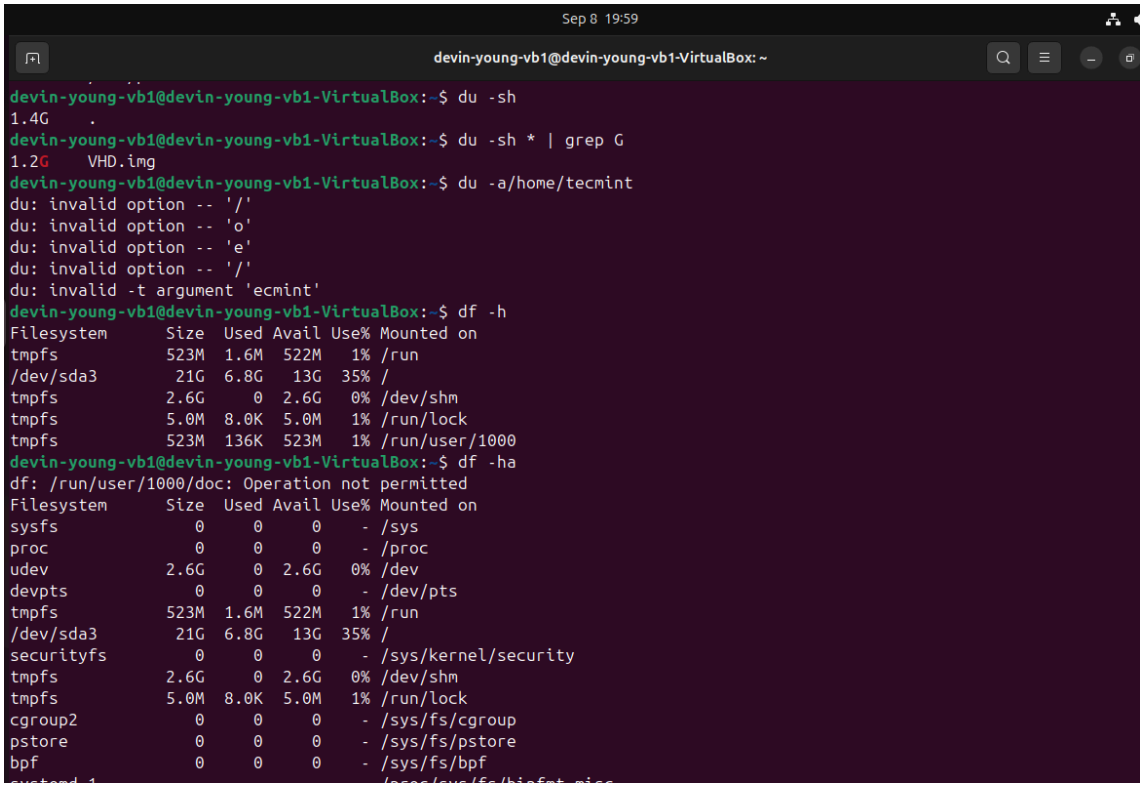
1. Identify the disk usage of various directories using the `du -sh` command.

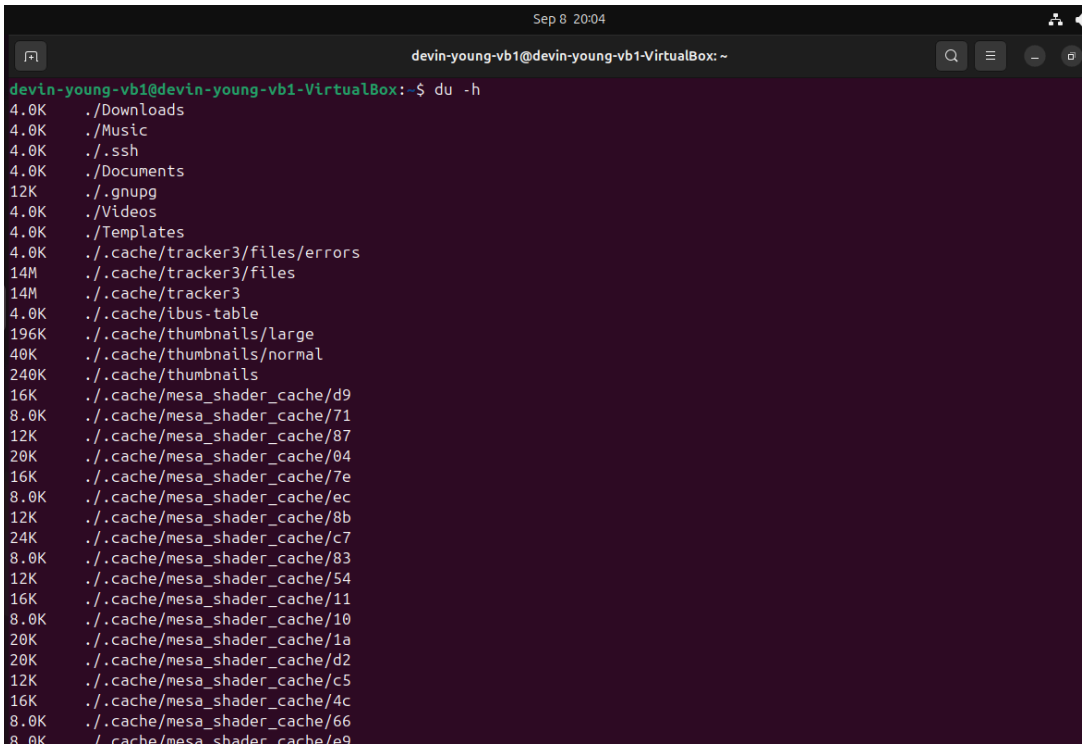
2. Find the top 5 largest files and directories on the system using the `find` command combined with `sort`.

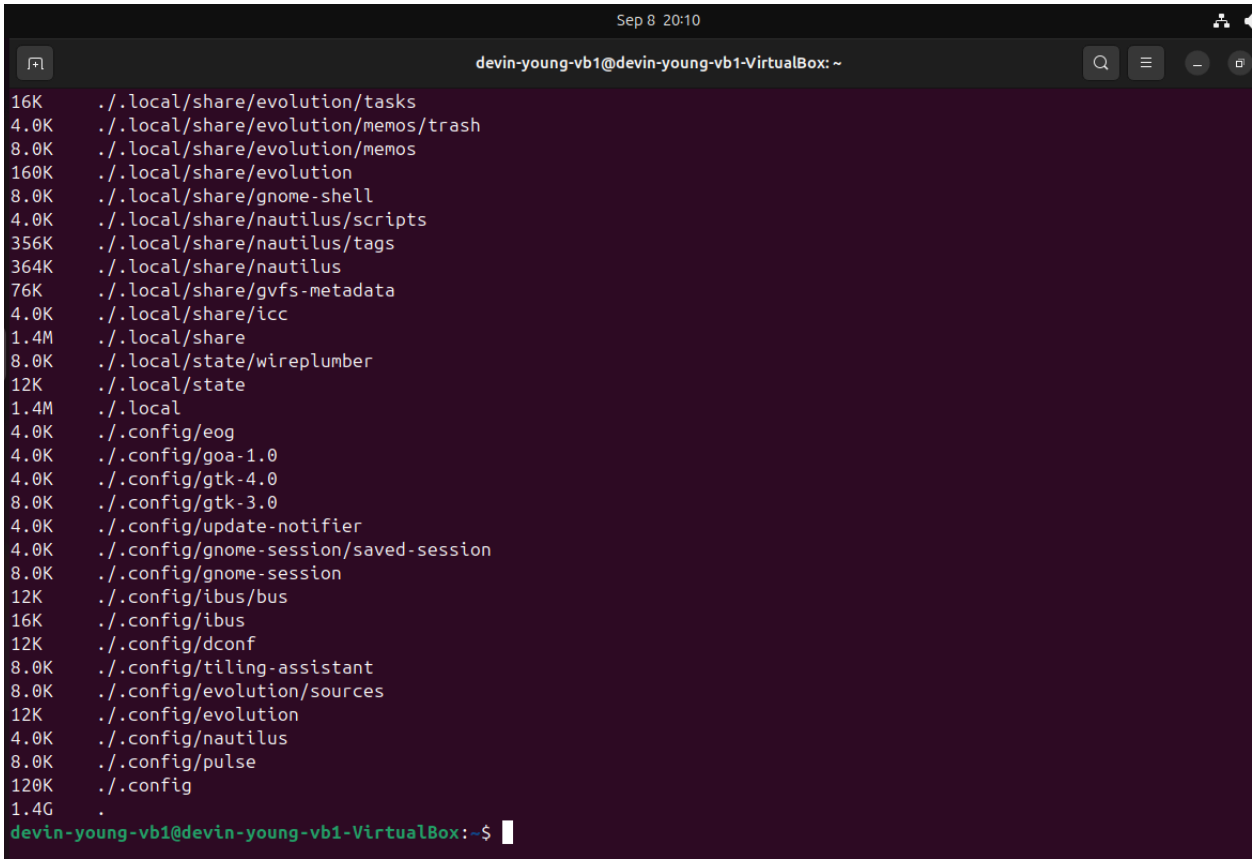
3. Clean up unnecessary files and directories (e.g., old log files, temporary files) to free up disk space.

4. Document the actions taken and the amount of space freed.

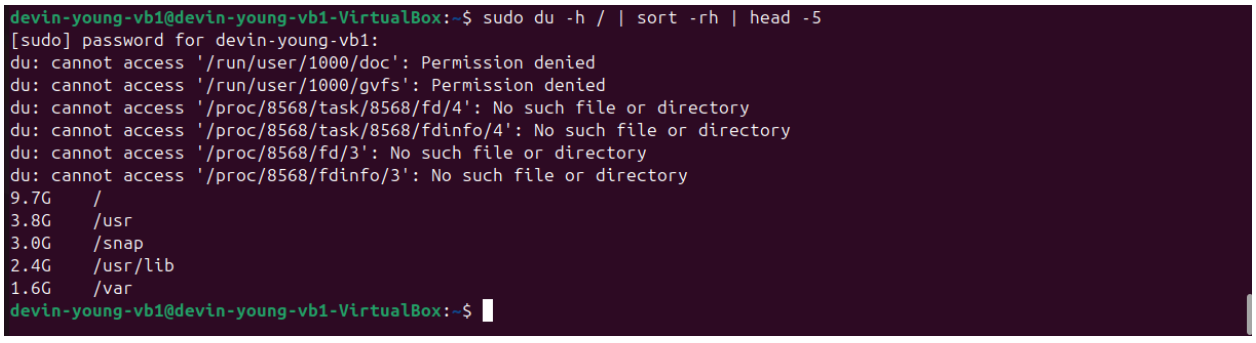
Step 1 - Identifying the disk usage of various directories using the du -sh command, and du -h command (which shows the disk usage for all directories and subdirectories)



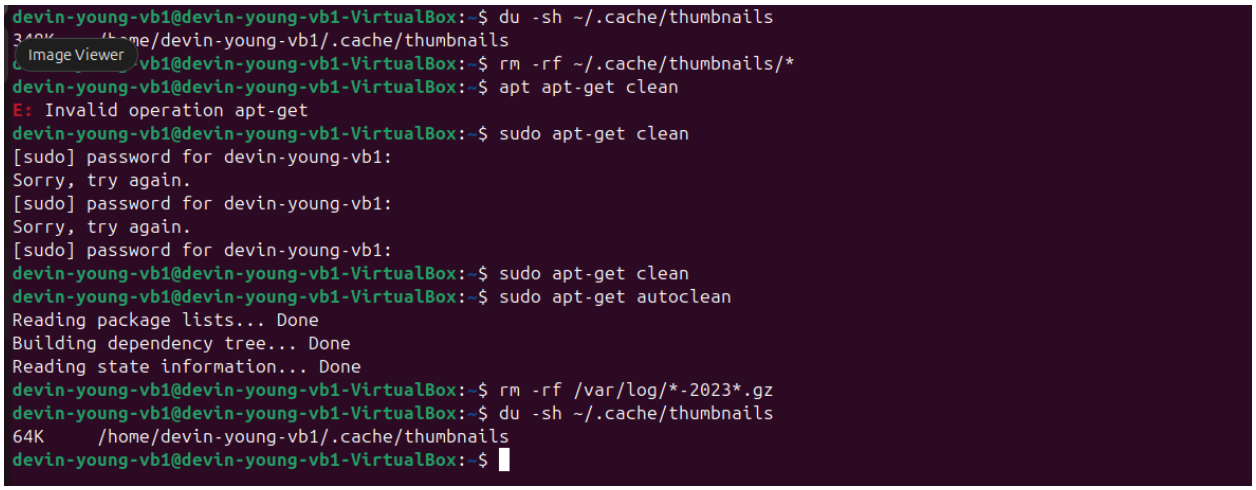




Step 2 - Top 5 largest files or directories on the system



Step 3 - Removing outdated archive logs, clearing local repository of package files that are left in /var/cache (over 300k of cache was removed)



Task 4: Resizing Filesystems

1. Create a logical volume using `LVM` (Logical Volume Manager).

2. Format the logical volume with a filesystem (e.g., ext4).

3. Mount the logical volume to a directory of your choice (e.g., `/mnt/lvtest`).

4. Add more space to the logical volume using `lvextend` and resize the filesystem using `resize2fs` or `xfs_growfs`.

5. Verify that the filesystem has been resized by checking the available space.

6. Document the entire process, including the commands used and the changes observed.

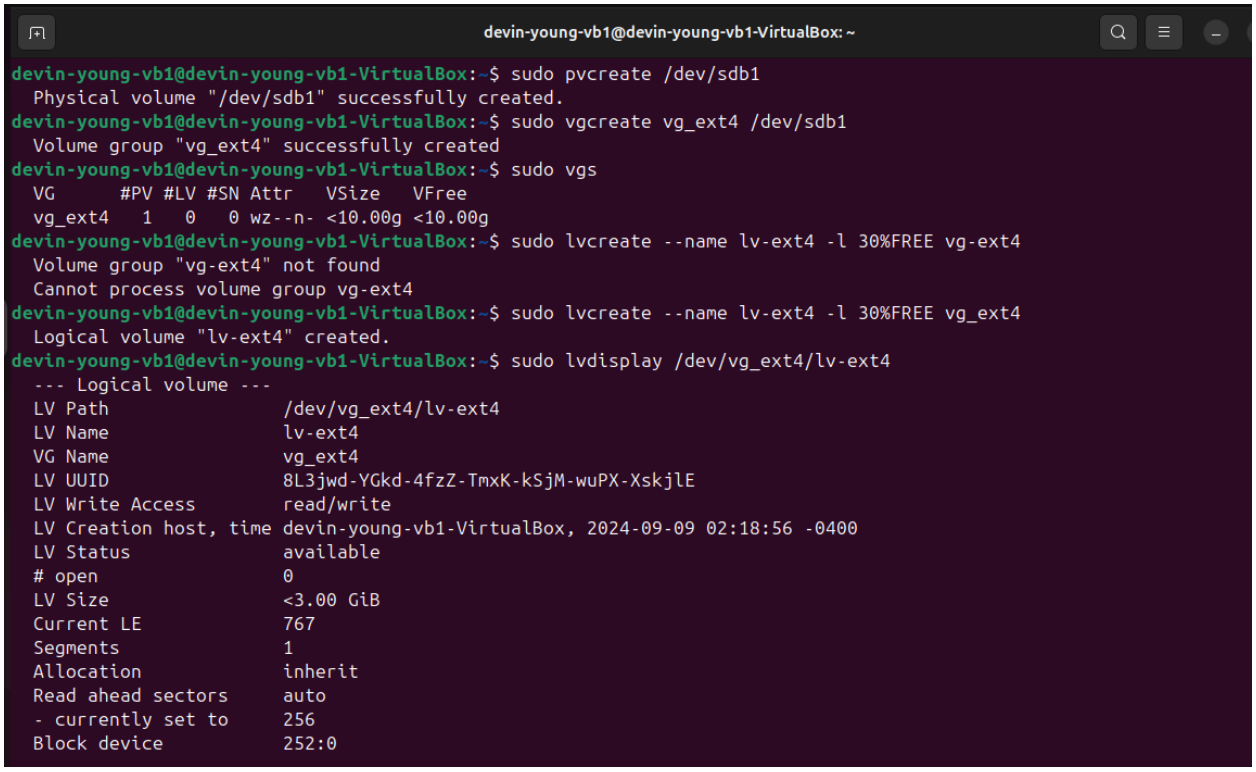
Step 1 - Preparing to create a logical volume (making sure I have what needs to be installed, and creating a partition so I can convert it to a logical volume)

```
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo apt-get install lvm2 rsync
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lvm2 is already the newest version (2.03.16-3ubuntu3.1).
rsync is already the newest version (3.2.7-1ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo fdisk /dev/sdb

Welcome to fdisk (util-linux 2.39.3).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

The device contains 'LVM2_member' signature and it will be removed by a write command. See fdisk(8) man page an
d --wipe option for more details.

Device does not contain a recognized partition table.
Created a new DOS (MBR) disklabel with disk identifier 0x66243138.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-52428799, default 2048): 2048
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-52428799, default 52428799): +10G
```

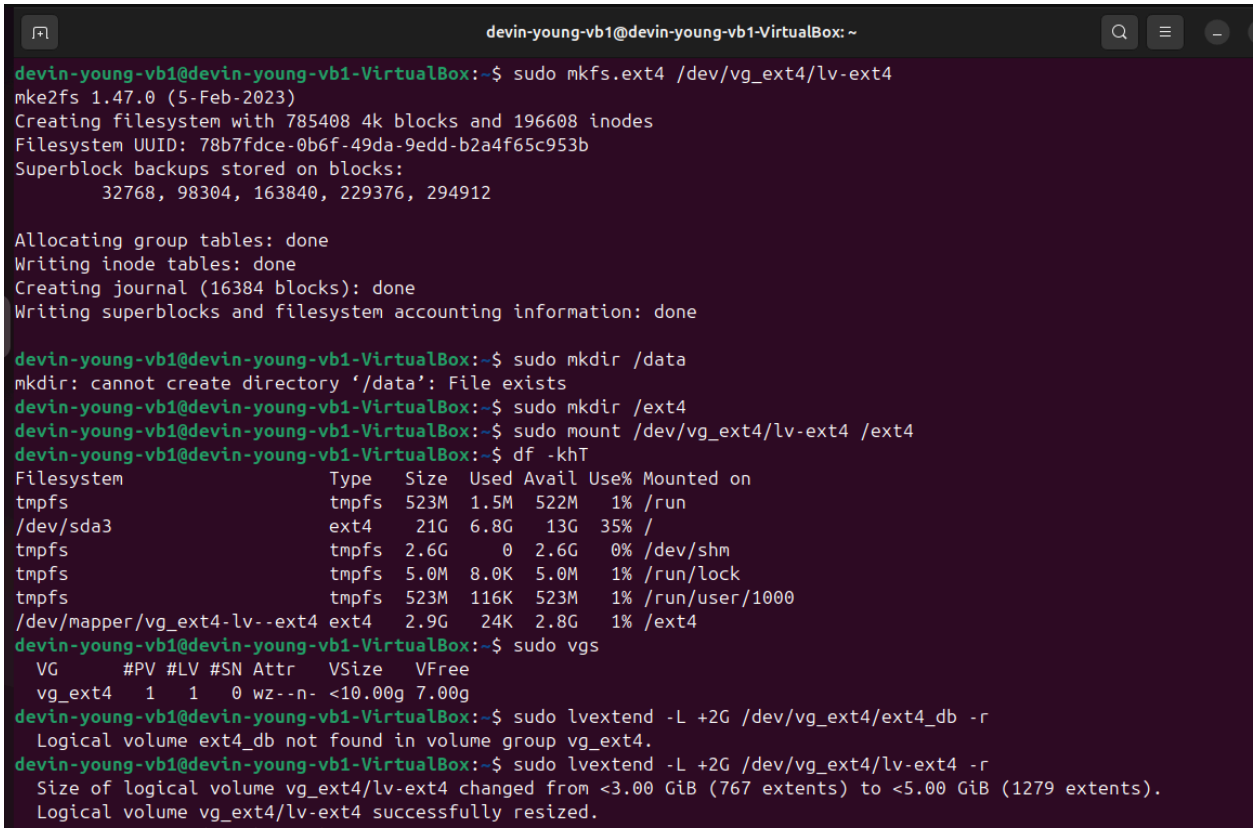
```
devin-young-vb1@devin-young-vb1-VirtualBox: ~

The device contains 'LVM2_member' signature and it will be removed by a write command. See fdisk(8) man page an
d --wipe option for more details.

Device does not contain a recognized partition table.
Created a new DOS (MBR) disklabel with disk identifier 0x66243138.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-52428799, default 2048): 2048
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-52428799, default 52428799): +10G

Created a new partition 1 of type 'Linux' and of size 10 GiB.

Command (m for help): t
Selected partition 1
Hex code or alias (type L to list all): 8e
Changed type of partition 'Linux' to 'Linux LVM'.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

Step 2 - Confirming the creation of the logical volume partition

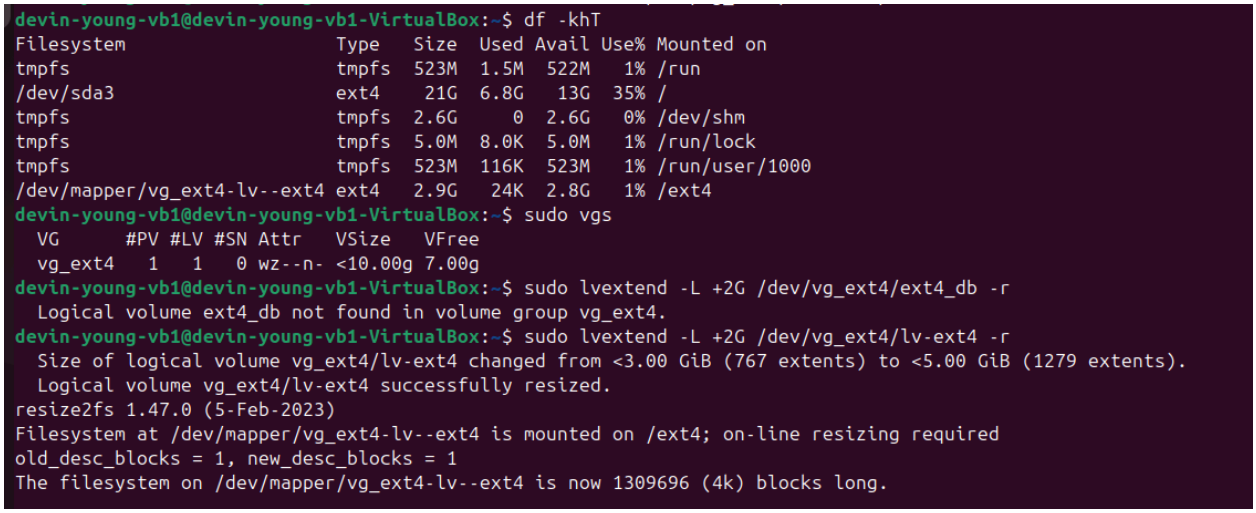
Step 3 - Creating the physical volume, volume group, and logical volume



Step 4 - Formatted the logical volume with a filesystem and created a new directory to mount the logical volume to it



Step 5 - Added more space to the logical volume by using the lvextend command (had trouble resizing the fileystem with the resize2fs command)



```
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo resize2fs /dev/sdb1
resize2fs 1.47.0 (5-Feb-2023)
resize2fs: Device or resource busy while trying to open /dev/sdb1
Couldn't find valid filesystem superblock.
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo resize2fs /dev/sdb1 SIZE
resize2fs 1.47.0 (5-Feb-2023)
resize2fs: Device or resource busy while trying to open /dev/sdb1
Couldn't find valid filesystem superblock.
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo resize2fs ext4 SIZE
resize2fs 1.47.0 (5-Feb-2023)
open: No such file or directory while opening ext4
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo resize2fs /dev/mapper/vg_ext4-lv-ext4 SIZE
resize2fs 1.47.0 (5-Feb-2023)
open: No such file or directory while opening /dev/mapper/vg_ext4-lv-ext4
```

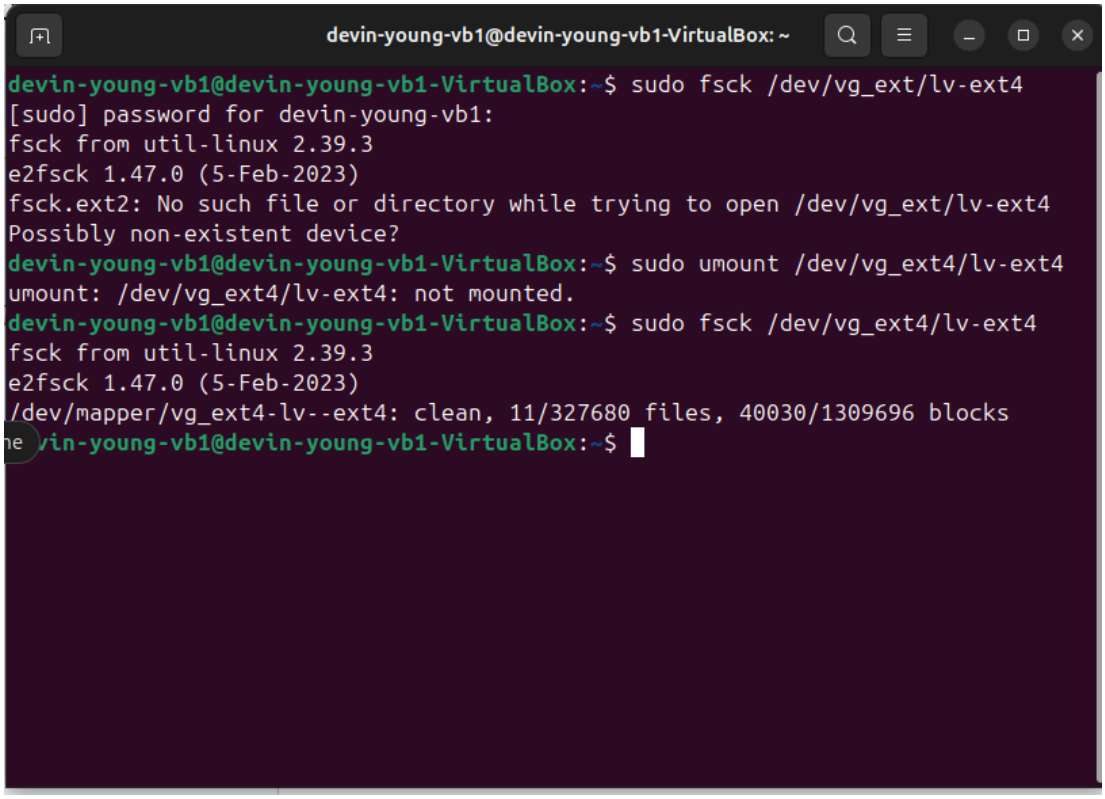
Task 5: Filesystem Health Check and Repair

1. Perform a filesystem check on an unmounted filesystem using the `fsck` command.

2. Simulate a filesystem issue by creating a corrupted file and running `fsck` to repair it.

3. Document the process, including the before-and-after state of the filesystem.

Step 1 - Performing a filesystem check on an unmounted fileystsem



Step 2 & 3 - Had trouble simulating a corrupted file but I understand what the fsck command does

```
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo fsck.ext4 -f disk.img
[sudo] password for devin-young-vb1:
e2fsck 1.47.0 (5-Feb-2023)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
disk.img: 11/12800 files (9.1% non-contiguous), 1837/12800 blocks
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo ls /mnt/testfs
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo cat /mnt/testfs/testfile.tx
cat: /mnt/testfs/testfile.txt: No such file or directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo dd if=/dev/zero of=disk.img
s=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.882592 s, 59.4 MB/s
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo mkfs.ext4 disk.img
mke2fs 1.47.0 (5-Feb-2023)
Discarding device blocks: done
```

```
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.882592 s, 59.4 MB/s
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo mkfs.ext4 disk.img
mke2fs 1.47.0 (5-Feb-2023)
Discarding device blocks: done
Creating filesystem with 12800 4k blocks and 12800 inodes

Allocating group tables: done
Writing inode tables: done
Creating journal (1024 blocks): done
Writing superblocks and filesystem accounting information: done

devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo mkdir /mnt/testfs
mkdir: cannot create directory ‘/mnt/testfs’: File exists
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo mkdir /mnt/testfiles
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo mount -o loop disk.img /mnt/t
estfiles
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo echo "This is a test file." |
 sudo tee /mnt/testfiles/testfile.txt
This is a test file.
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo dd if=/dev/urandom of=disk.im
g bs=13 seek=4 conv=notrunc
```

Part 3: Filesystem Security and Permissions

Task 6: Managing Permissions and Ownership

1. Create a new directory and file within your filesystem.

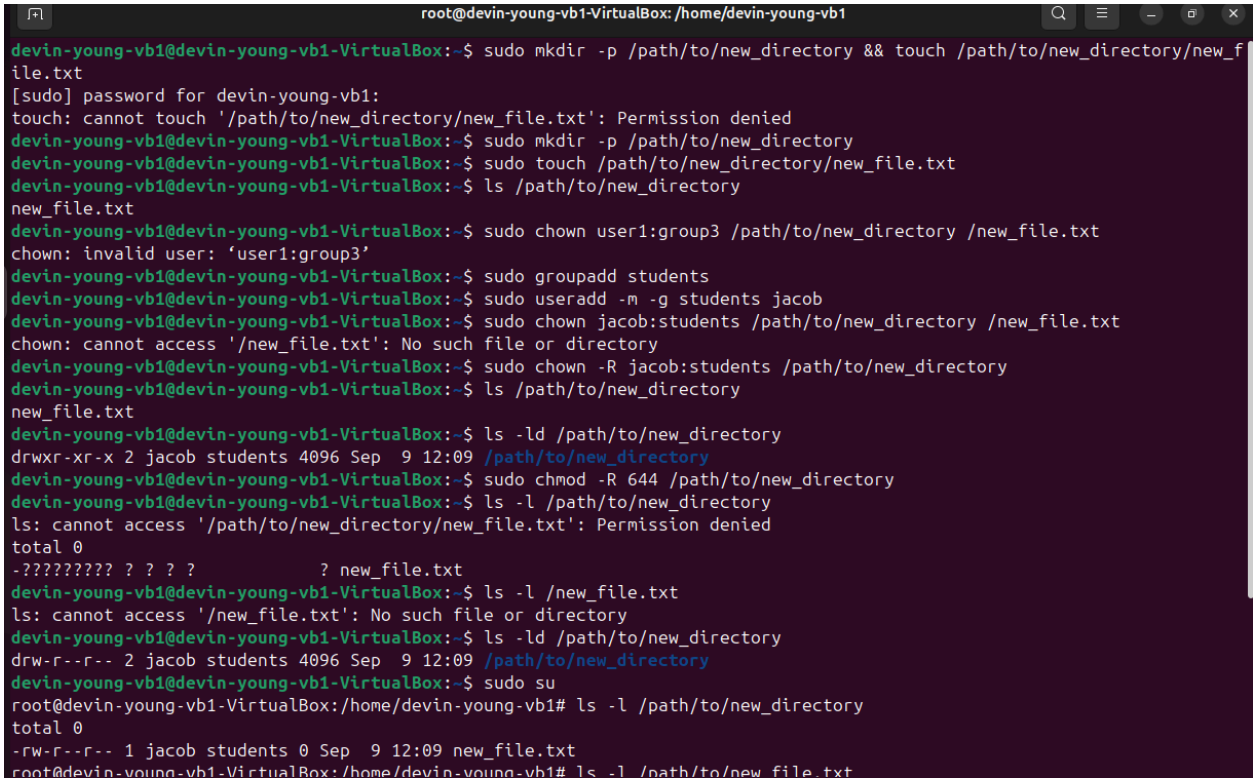
2. Change the ownership of the directory and file to a specific user and group using `chown`.

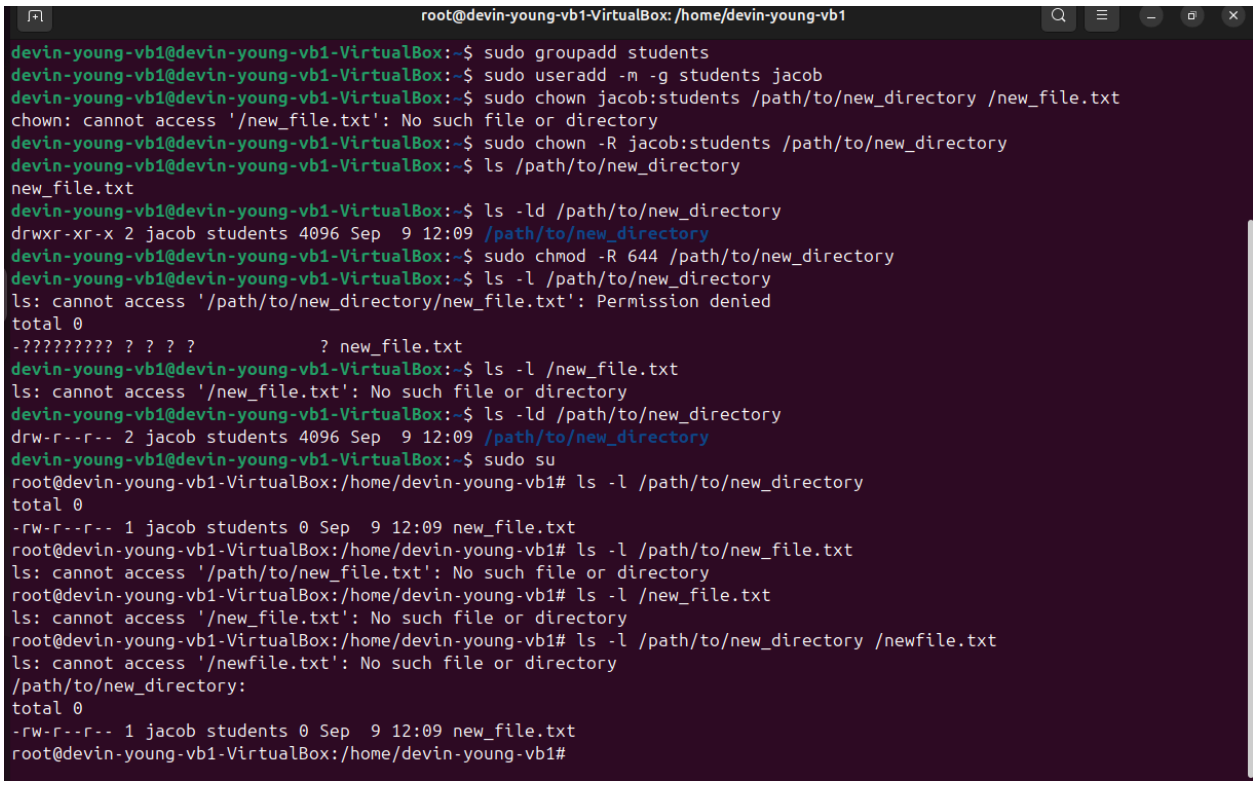
3. Modify the permissions of the directory and file using `chmod` to grant specific access rights (e.g., read-only, execute).

4. Verify the changes using the `ls -l` command.

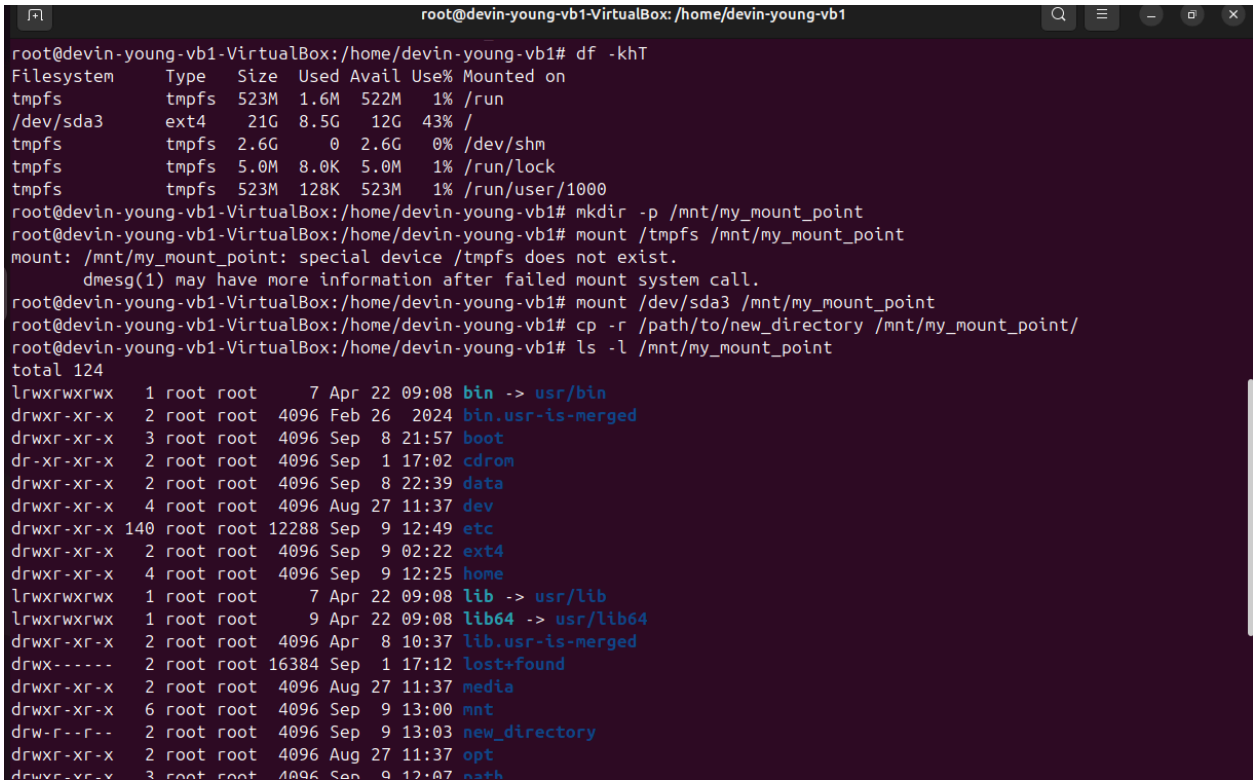
5. Document the commands used and the resulting permissions.

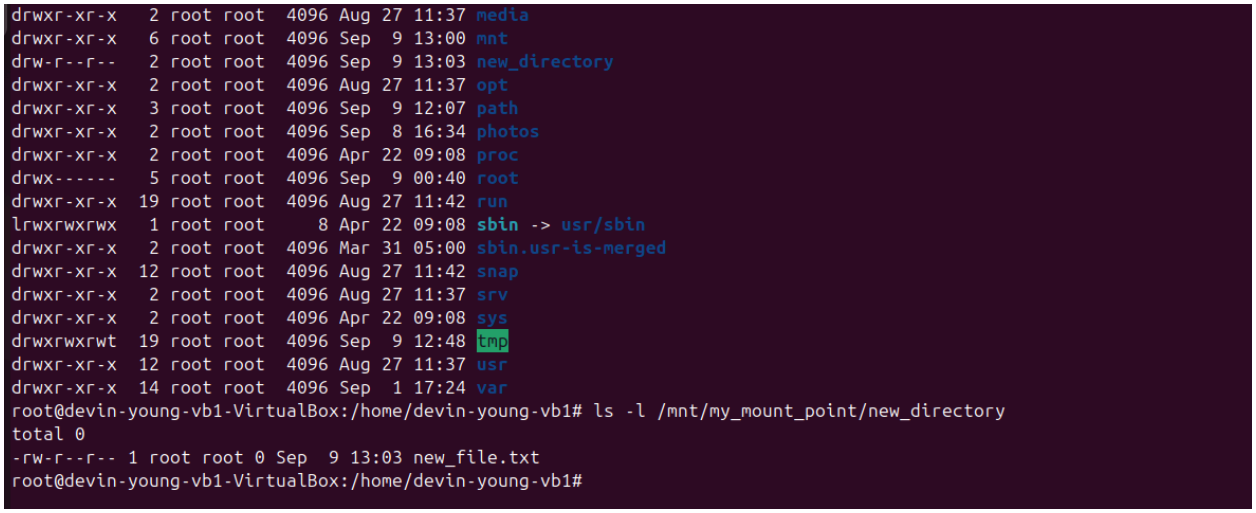
Step (1-4) - In the first two screenshots I created a directory called “new directory” and a file called “new_file.txt”, changed the ownership of the directory and file to a specific user and group I created (“students” being the group, “jacob” being the user), modified the permissions so that the file owner had permission to read and write, while the group and world permission could only read (-rw-r–r or 644), and verified the changes using the ls -l command.


root@devin-young-vb1-VirtualBox: /home/devin-young-vb1
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo mkdir -p /path/to/new_directory && touch /path/to/new_directory/new_file.txt
[sudo] password for devin-young-vb1:
touch: cannot touch '/path/to/new_directory/new_file.txt': Permission denied
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo mkdir -p /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo touch /path/to/new_directory/new_file.txt
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls /path/to/new_directory
new_file.txt
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo chown user1:group3 /path/to/new_directory /new_file.txt
chown: invalid user: ‘user1:group3’
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo groupadd students
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo useradd -m -g students jacob
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo chown jacob:students /path/to/new_directory /new_file.txt
chown: cannot access '/new_file.txt': No such file or directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo chown -R jacob:students /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls /path/to/new_directory
new_file.txt
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -ld /path/to/new_directory
drwxr-xr-x 2 jacob students 4096 Sep 9 12:09 /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo chmod -R 644 /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -l /path/to/new_directory
ls: cannot access '/path/to/new_directory/new_file.txt': Permission denied
total 0
-????????? ? ? ? ?   ? new_file.txt
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -l /new_file.txt
ls: cannot access '/new_file.txt': No such file or directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -ld /path/to/new_directory
drw-r--r-- 2 jacob students 4096 Sep 9 12:09 /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo su
root@devin-young-vb1-VirtualBox:/home/devin-young-vb1# ls -l /path/to/new_directory
total 0
-rw-r--r-- 1 jacob students 0 Sep 9 12:09 new_file.txt


root@devin-young-vb1-VirtualBox: /home/devin-young-vb1
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo groupadd students
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo useradd -m -g students jacob
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo chown jacob:students /path/to/new_directory /new_file.txt
chown: cannot access '/new_file.txt': No such file or directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo chown -R jacob:students /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls /path/to/new_directory
new_file.txt
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -ld /path/to/new_directory
drwxr-xr-x 2 jacob students 4096 Sep 9 12:09 /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo chmod -R 644 /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -l /path/to/new_directory
ls: cannot access '/path/to/new_directory/new_file.txt': Permission denied
total 0
-????????? ? ? ? ?   ? new_file.txt
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -l /new_file.txt
ls: cannot access '/new_file.txt': No such file or directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ ls -ld /path/to/new_directory
drw-r--r-- 2 jacob students 4096 Sep 9 12:09 /path/to/new_directory
devin-young-vb1@devin-young-vb1-VirtualBox:~$ sudo su
root@devin-young-vb1-VirtualBox:/home/devin-young-vb1# ls -l /path/to/new_directory
total 0
-rw-r--r-- 1 jacob students 0 Sep 9 12:09 new_file.txt
root@devin-young-vb1-VirtualBox:/home/devin-young-vb1# ls -l /path/to/new_file.txt
ls: cannot access '/path/to/new_file.txt': No such file or directory
root@devin-young-vb1-VirtualBox:/home/devin-young-vb1# ls -l /new_file.txt
ls: cannot access '/new_file.txt': No such file or directory
root@devin-young-vb1-VirtualBox:/home/devin-young-vb1# ls -l /path/to/new_directory /newfile.txt
ls: cannot access '/newfile.txt': No such file or directory
/path/to/new_directory:
total 0
-rw-r--r-- 1 jacob students 0 Sep 9 12:09 new_file.txt
root@devin-young-vb1-VirtualBox:/home/devin-young-vb1#

Steps (1-4 Part II) - Lastly, I created a mount point and mounted a filesystem I have to that mount point, and then copied the directory to the mount point so the directory I created could be mounted to the file system. Afterwards, I also confirmed that the changes had been made.





Task 7: Set Up Disk Quotas

1. Enable disk quotas on a specific filesystem.

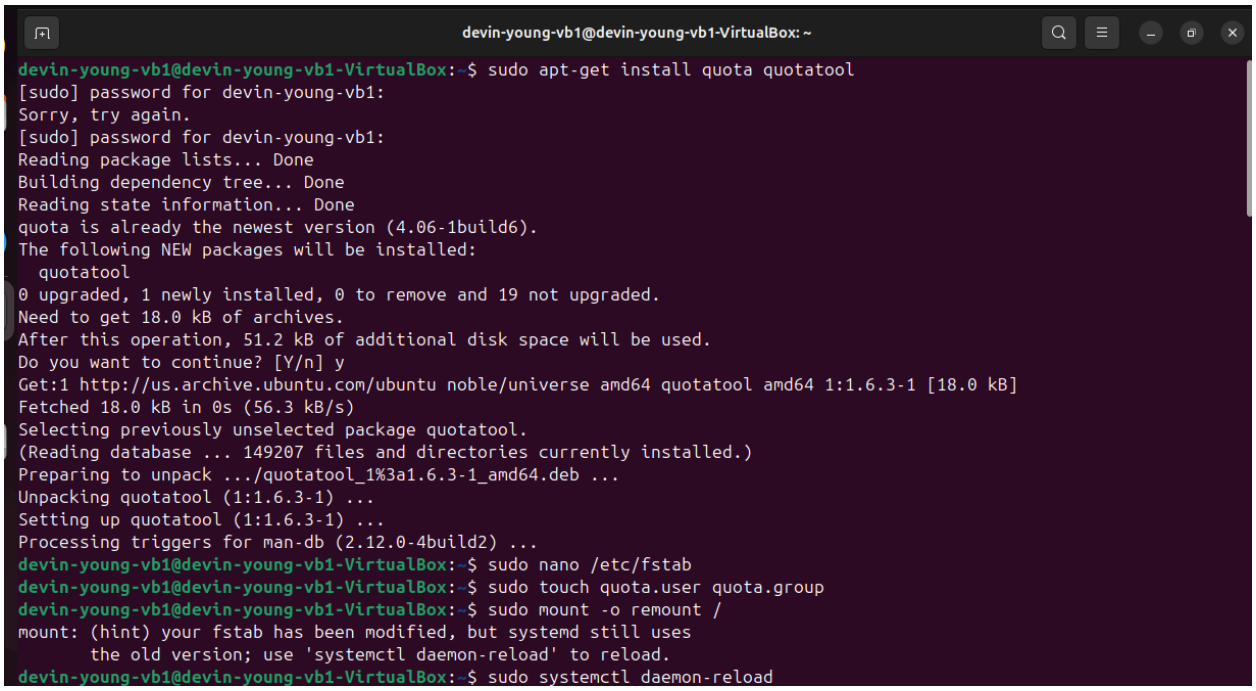
2. Set up user quotas to limit the disk space and inode usage for a specific user.

- Use the `edquota` command to set quotas.

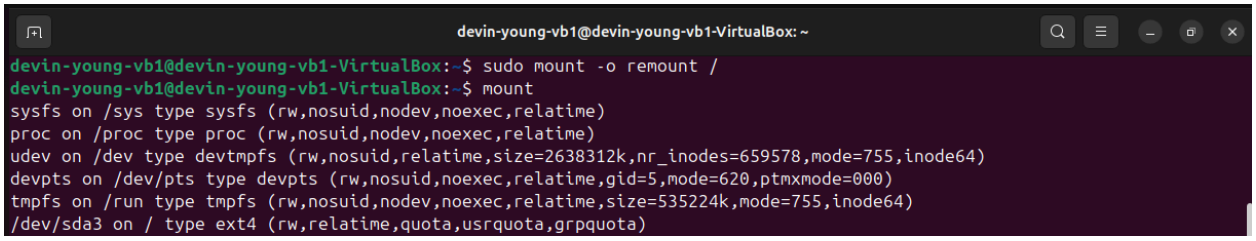
3. Test the quota by creating files as the user and observing when the quota is enforced.

4. Document the setup process and the results of the quota test.

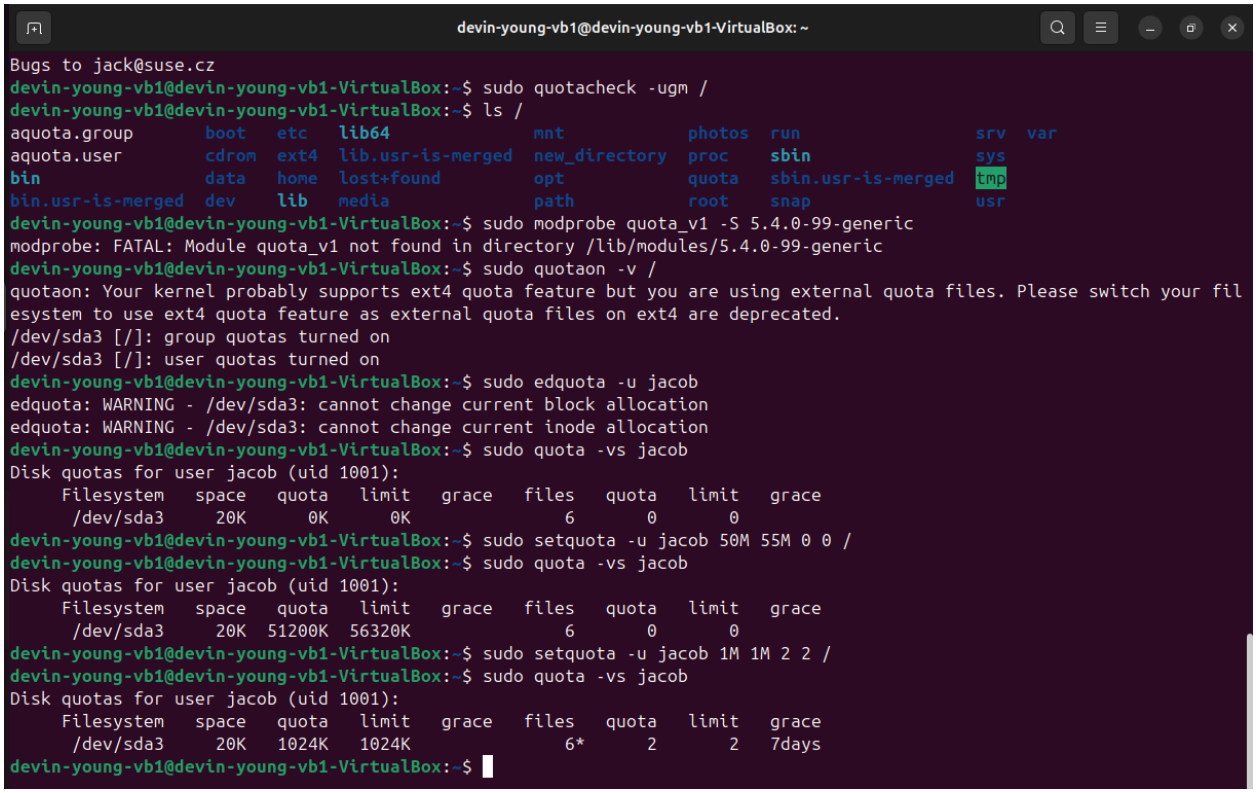
Step 1 - Installing the tools to perform the task



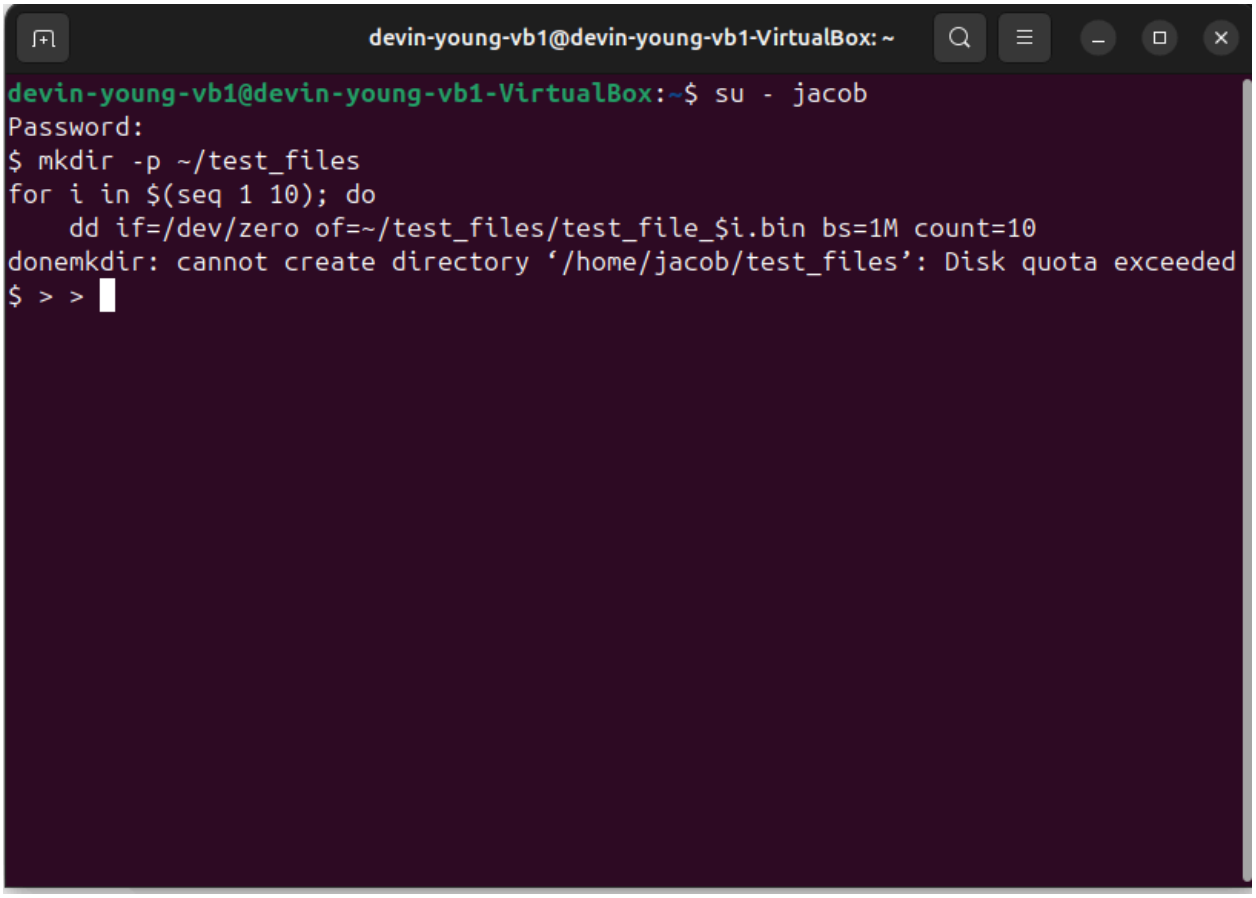
Step 2 - Confirming that the user quota and group quota have been included in the etc/fstab



Step 3 - Set up user quotas to the disk space and inodes for a specific user



Step 4 - Creating files as a user to test the quota (could not create 10 files because it exceeded the enforced quota, which was 6 files)



Reflection

I learned that there's a lot of backtracking and confirmation involved with Linux filesystem management. There were many times I had to confirm that the commands I entered were enforced, and many times where I had to trace my steps backwards to see what I missed, whether that was creating a directory, making sure I selected the correct path, checking to see if a filesystem I created is mounted or unmounted, verifying that I correctly formatted a filesystem, making sure I'm referencing the correct partition because it can be my logical or physical volume, etc. I also learned that documenting your steps as you manage a Linux filesystem is extremely important, especially since I took the route of using a Ubuntu virtual machine instead of my Ubuntu host OS. Meaning, without the real-time documentation, I would've been trying to

figure out what my previous steps were whenever I took a break and came back to working on this assignment at a later time. In addition, I learned that with many tasks, you just use different options/flags with the same commands. There were a few new commands, but most of the commands I used, I saw before. There were just different flags syntax to go along with them. Lastly, I learned that creating scripts are extremely helpful. I wouldn't have had to manually type out so many commands if I created scripts to do those commands. I don't believe that's something I could've done on this project considering how long that would take, but if I ever need to in the future that's definitely an efficient option.