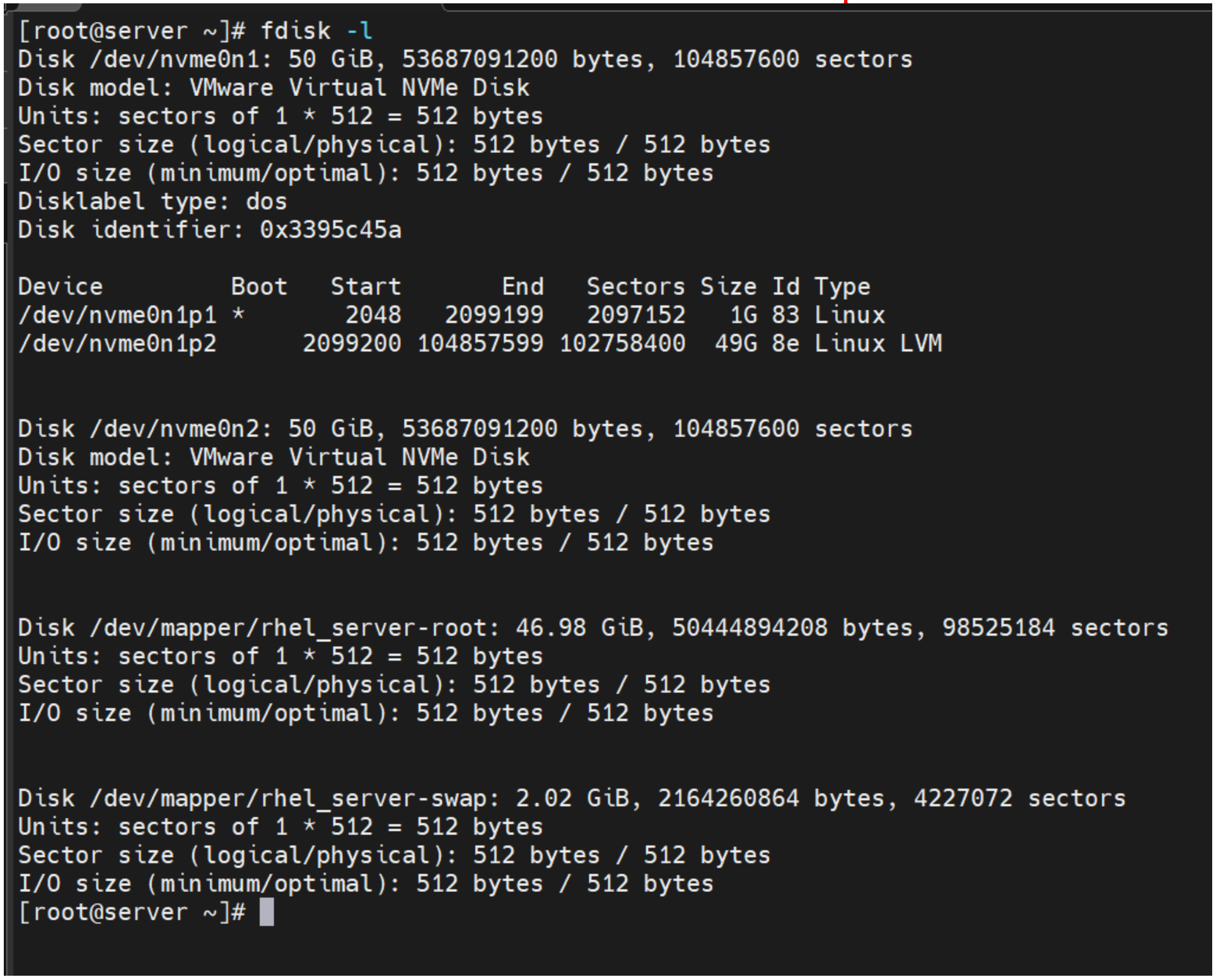
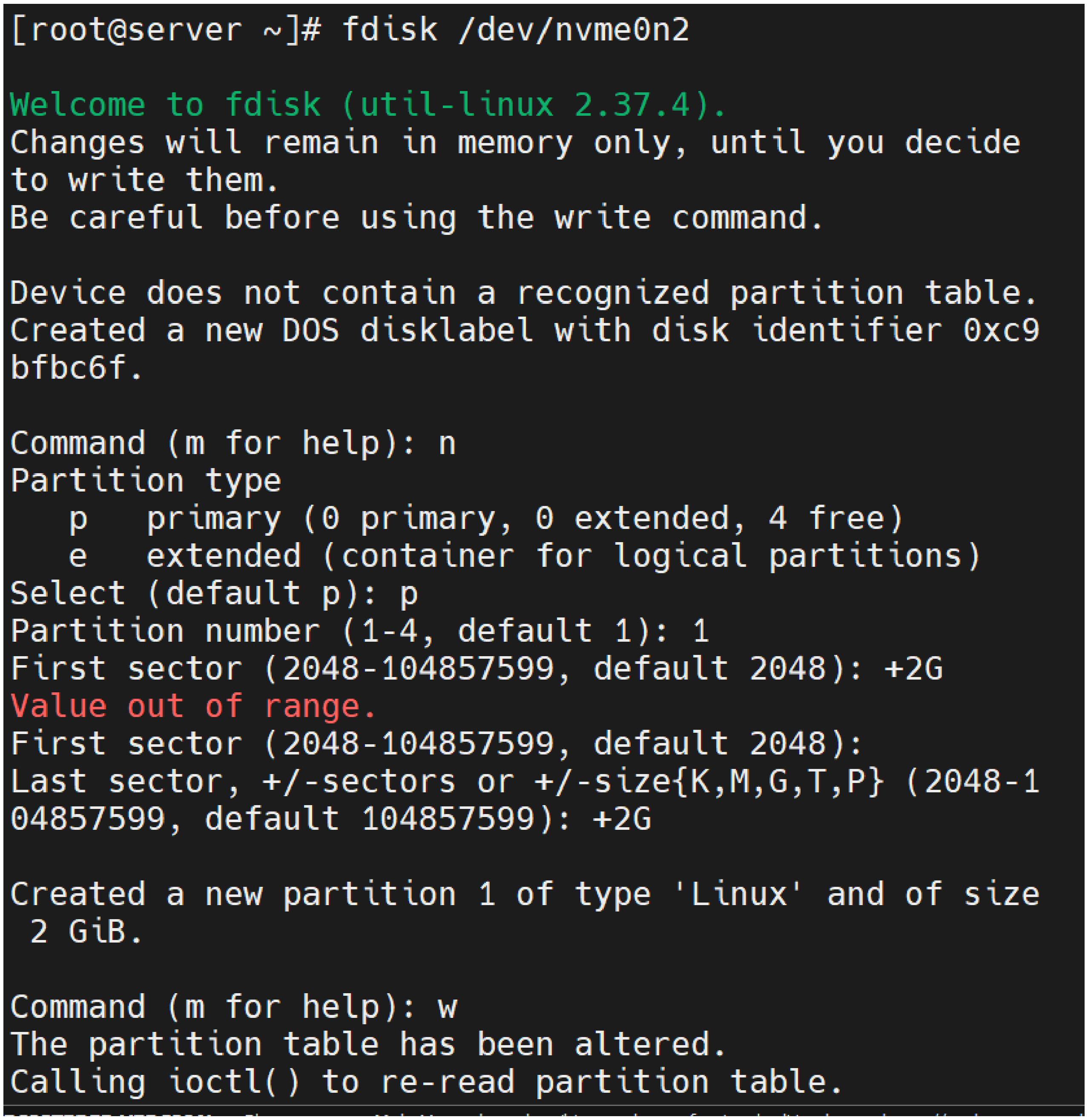
Lab 3

1. Use `fdisk -l` to locate information about the partition sizes.



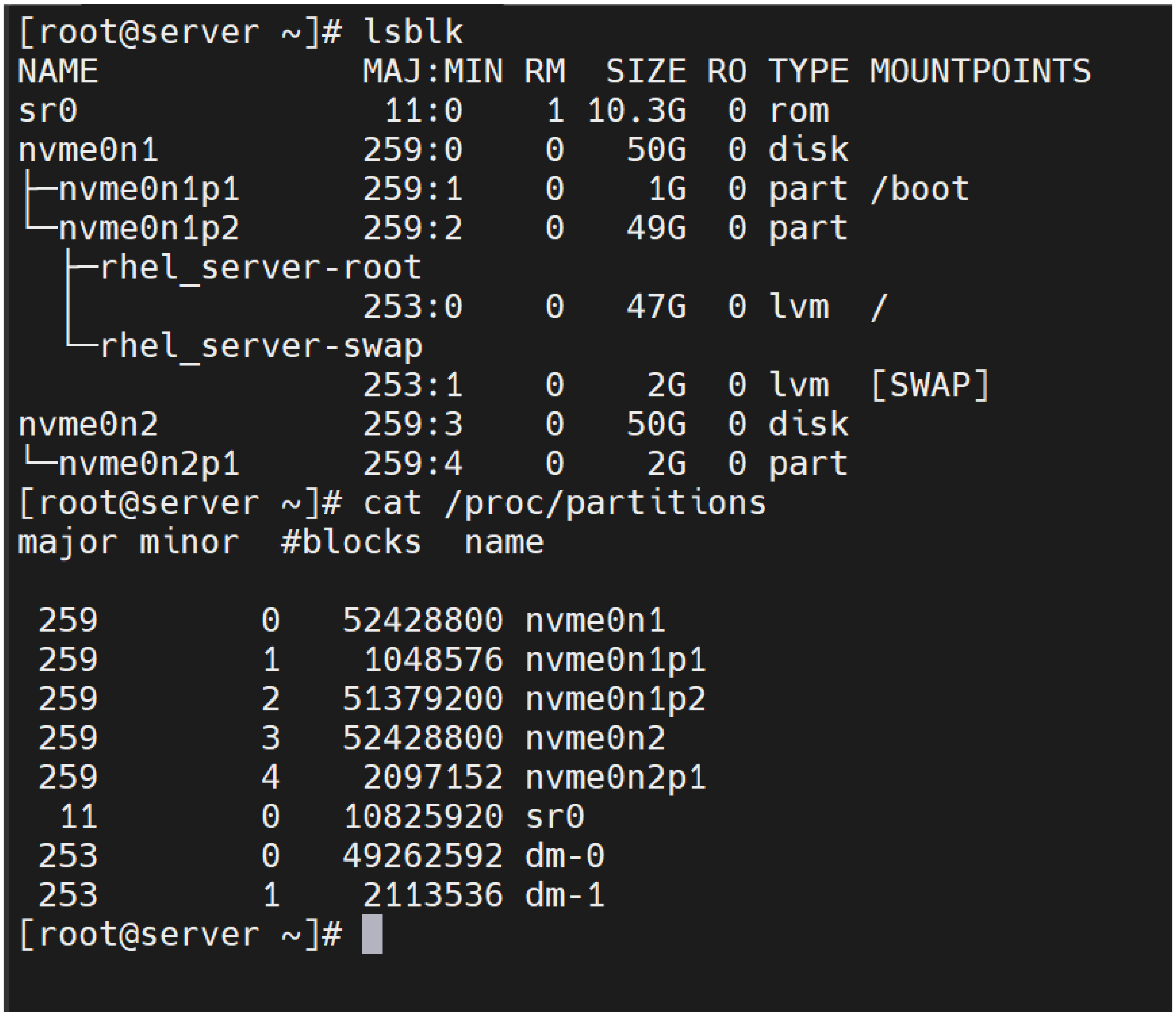
```
[root@server ~]# fdisk -l
Disk /dev/nvme0n1: 50 GiB, 53687091200 bytes, 104857600 sectors
Disk model: VMware Virtual NVMe Disk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x3395c45a

Device         Boot   Start       End   Sectors Size Id Type
/dev/nvme0n1p1 *       2048   2099199   2097152   1G 83 Linux
/dev/nvme0n1p2      2099200 104857599 102758400  49G 8e Linux LVM


Disk /dev/nvme0n2: 50 GiB, 53687091200 bytes, 104857600 sectors
Disk model: VMware Virtual NVMe Disk
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 /dev/mapper/rhel_server-root: 46.98 GiB, 50444894208 bytes, 98525184 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 /dev/mapper/rhel_server-swap: 2.02 GiB, 2164260864 bytes, 4227072 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
[root@server ~]#
```

2. Use `fdisk` to add a new logical partition that is 2GB in size.Fdisk /dev/sdb



```
[root@server ~]# fdisk /dev/nvme0n2

Welcome to fdisk (util-linux 2.37.4).
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.
Created a new DOS disklabel with disk identifier 0xc9
bfbc6f.

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 (2048-104857599, default 2048): +2G
Value out of range.
First sector (2048-104857599, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-1
04857599, default 104857599): +2G

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

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

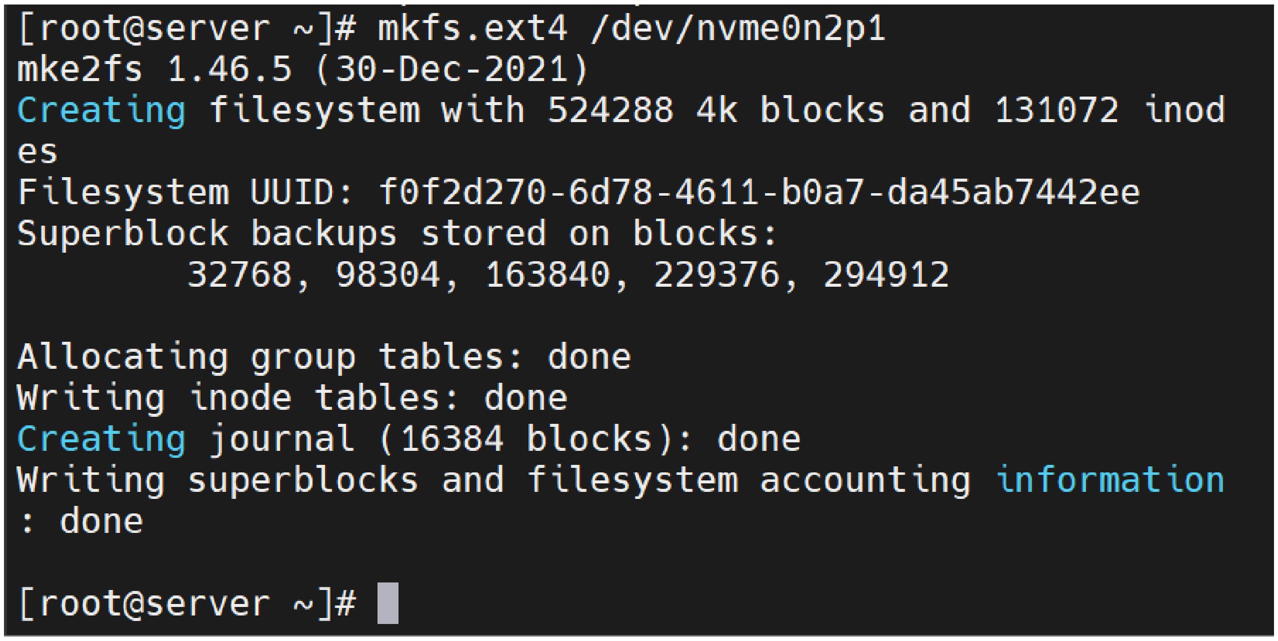
3. Did the kernel feel the changes? Display the content of /proc/partitions file? What didyou notice? How to overcome that?



```
[root@server ~]# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sr0                  11:0    1 10.3G  0 rom
nvme0n1             259:0    0   50G  0 disk
├─nvme0n1p1         259:1    0    1G  0 part /boot
└─nvme0n1p2         259:2    0   49G  0 part
  ├─rhel_server-root
  │                 253:0    0   47G  0 lvm  /
  └─rhel_server-swap
                    253:1    0    2G  0 lvm  [SWAP]
nvme0n2             259:3    0   50G  0 disk
└─nvme0n2p1         259:4    0    2G  0 part
[root@server ~]# cat /proc/partitions
major minor  #blocks  name

 259        0   52428800 nvme0n1
 259        1    1048576 nvme0n1p1
 259        2   51379200 nvme0n1p2
 259        3   52428800 nvme0n2
 259        4    2097152 nvme0n2p1
  11        0   10825920 sr0
 253        0   49262592 dm-0
 253        1    2113536 dm-1
[root@server ~]#
```

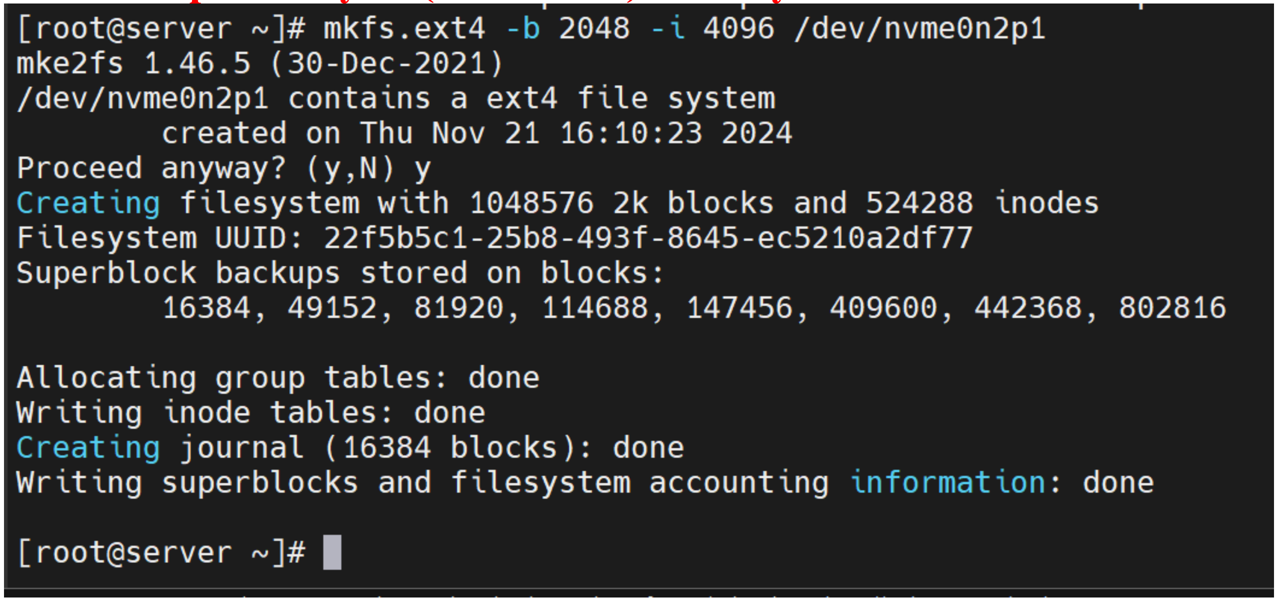
Yes feel the change and if else we can use partprobe

4. Make a new `ext4` file system on the new logical partition you just created.



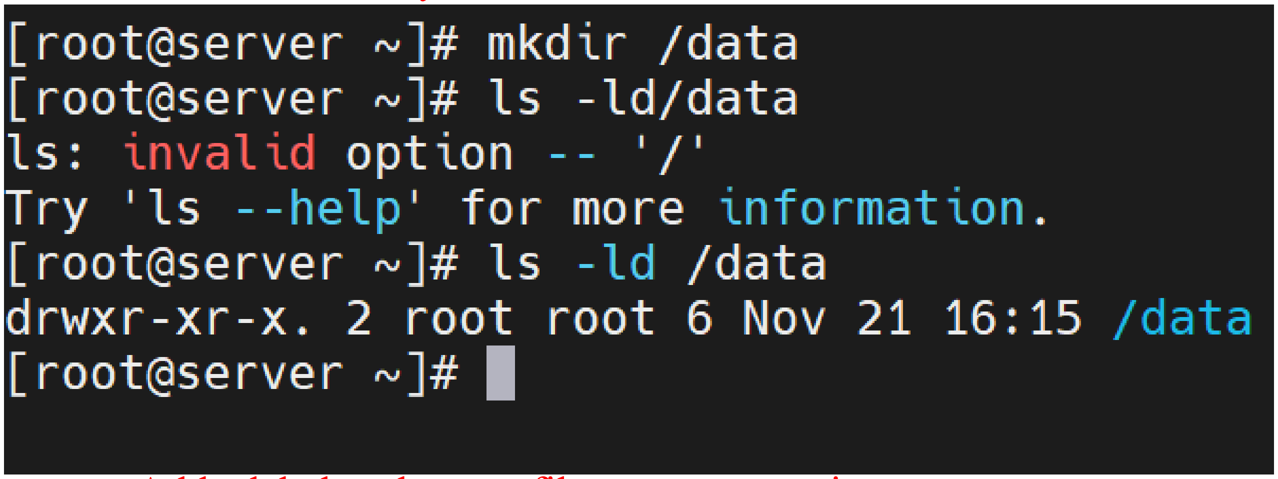
```
[root@server ~]# mkfs.ext4 /dev/nvme0n2p1
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 524288 4k blocks and 131072 inod
es
Filesystem UUID: f0f2d270-6d78-4611-b0a7-da45ab7442ee
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information
: done

[root@server ~]#
```

Bonus: Try creating the `ext4` filesystem with 2k blocks and one inode per every 4k(two blocks) of filesystem.



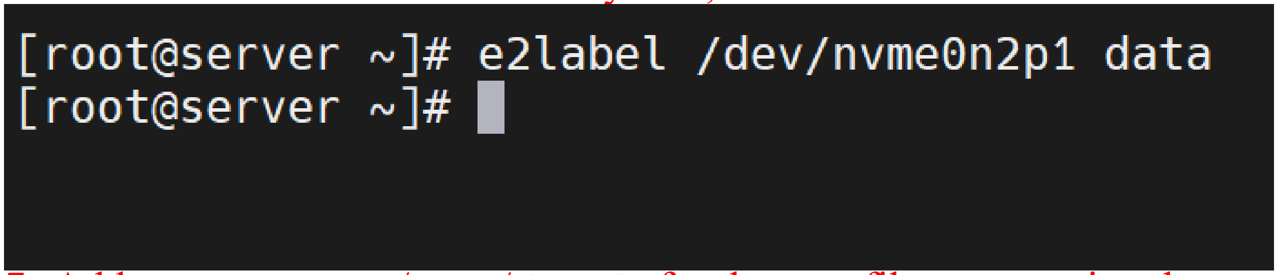
```
[root@server ~]# mkfs.ext4 -b 2048 -i 4096 /dev/nvme0n2p1
mke2fs 1.46.5 (30-Dec-2021)
/dev/nvme0n2p1 contains a ext4 file system
        created on Thu Nov 21 16:10:23 2024
Proceed anyway? (y,N) y
Creating filesystem with 1048576 2k blocks and 524288 inodes
Filesystem UUID: 22f5b5c1-25b8-493f-8645-ec5210a2df77
Superblock backups stored on blocks:
        16384, 49152, 81920, 114688, 147456, 409600, 442368, 802816

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

[root@server ~]#
```

5. Create a directory, name it `/data`.



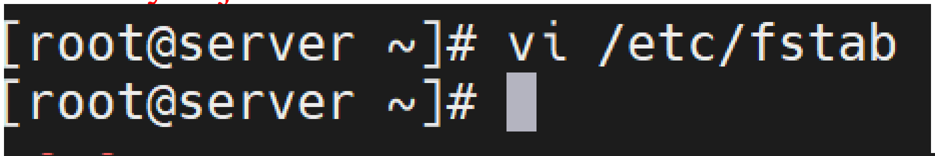
```
[root@server ~]# mkdir /data
[root@server ~]# ls -ld/data
ls: invalid option -- '/'
Try 'ls --help' for more information.
[root@server ~]# ls -ld /data
drwxr-xr-x. 2 root root 6 Nov 21 16:15 /data
[root@server ~]#
```

6. Add a label to the new filesystem, name it `data`.

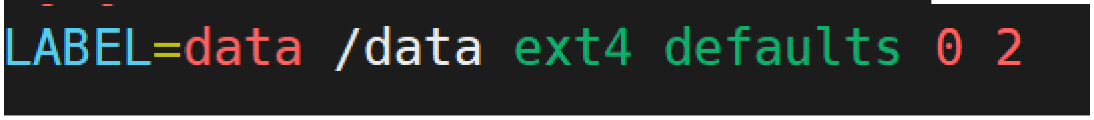


```
[root@server ~]# e2label /dev/nvme0n2p1 data
[root@server ~]#
```

7. Add a new entry to `/etc/fstab` for the new filesystem using the label you just create.

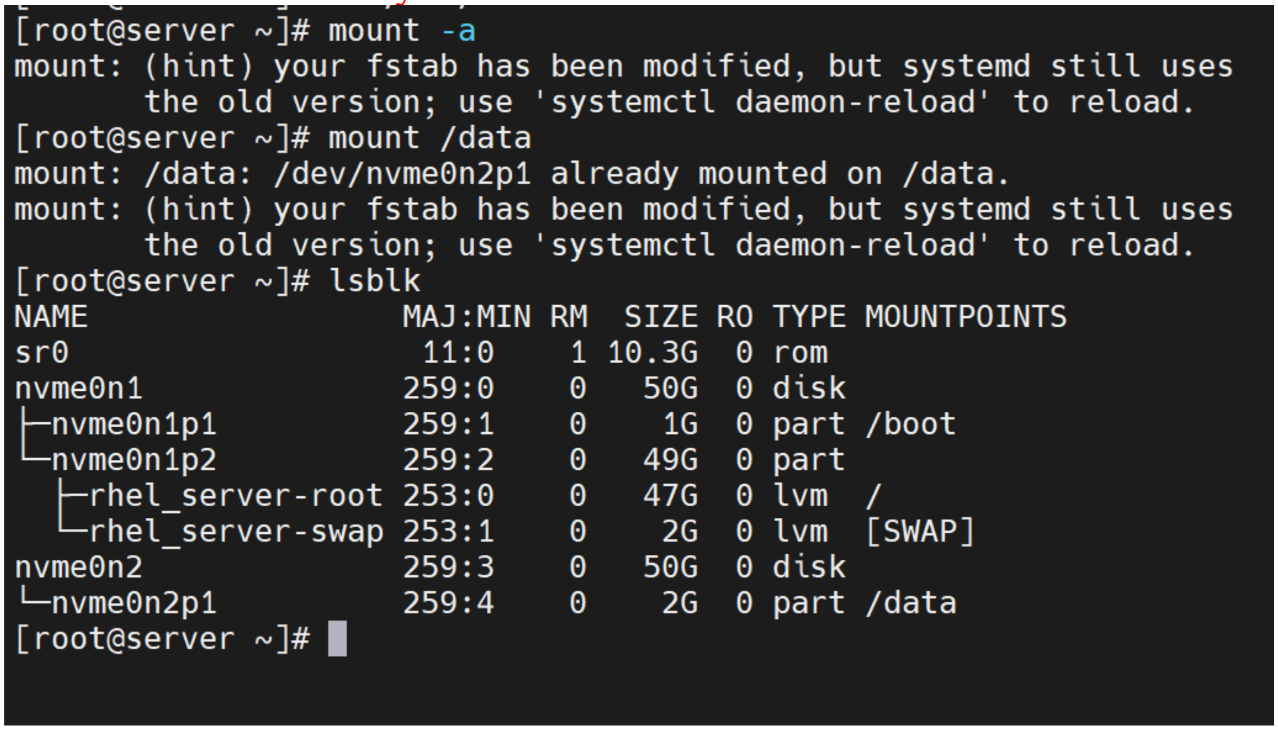


```
[root@server ~]# vi /etc/fstab
[root@server ~]#
```



```
LABEL=data /data ext4 defaults 0 2
```

8. Mount the new filesystem



```
[root@server ~]# mount -a
mount: (hint) your fstab has been modified, but systemd still uses
       the old version; use 'systemctl daemon-reload' to reload.
[root@server ~]# mount /data
mount: /data: /dev/nvme0n2p1 already mounted on /data.
mount: (hint) your fstab has been modified, but systemd still uses
       the old version; use 'systemctl daemon-reload' to reload.
[root@server ~]# lsblk
NAME                    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sr0                      11:0    1 10.3G  0 rom
nvme0n1                 259:0    0   50G  0 disk
├─nvme0n1p1             259:1    0    1G  0 part /boot
└─nvme0n1p2             259:2    0   49G  0 part
  ├─rhel_server-root 253:0    0   47G  0 lvm  /
  └─rhel_server-swap 253:1    0    2G  0 lvm  [SWAP]
nvme0n2                 259:3    0   50G  0 disk
└─nvme0n2p1             259:4    0    2G  0 part /data
[root@server ~]#
```

9. Display your swap size.

10. Create a swap file of size 512MB.
11. Add the swap file to the virtual memory of the system.
12. Display the swap size.
13. Use the `fdisk` command to create 2 Linux LVM (0x8e) partitions using "unpartitioned" space on your hard disk. These partitions should all be the same size; to speed up the lab, do not make them larger than 300 MB each. Make sure to write the changes to disk by using the w command to exit the fdisk utility. Run the partprobe command after exiting the fdisk utility.

14. Initialize your Linux LVM partitions as physical volumes with the pvcreate command. You can use the pvdisplay command to verify that the partitions have been initialized as physical volumes.
15. Using only one of your physical volumes, create a volume group called test0. Use the vgdisplay command to verify that the volume group was created.

16. Create a small logical volume (LV) called data that uses about 30 percent of the available space of the test0 volume group. Look for VG Size and Free PE/Size in the output of the vgdisplay command to assist you with this. Use the lvdisplay command to verify your work.

17. Create an xfs filesystem on your new LV.
18. Make a new directory called /data and then mount the new LV under the /data directory. Create a "large file" in this volume.

19. Enlarge the LV that you created in Sequence 1 (/dev/test0/data) by using approximately 25 percent of the remaining free space in the test0 volume group. Then, enlarge the filesystem of the LV.

20. Verify that the file /data/bigfile still exists in the LV. Run the df command and check to verify that more free disk space is now available on the LV.

21. Use the remaining extents in the test0 volume group to create a second LV called docs.

22. Run the vgdisplay command to verify that there are no free extents left in the test0 volume group.

23. Create an xfs filesystem on the new LV, make a mount point called /docs and mount the docs LV using this mount point.

24. Add all of the remaining unused physical volumes that you created in Sequence 1 to the test0 volume group.

25. If you run vgdisplay again, there now should be free extents (provided by the new physical volumes) in the test0 volume group. Extend the docs LV and underlying filesystem to make use of all of the free extents of the test0 volume group. Verify your actions.