End Lab 00:38:03

External IP address

35.202.32.112

username

student-04-c4f18f86210e

Download PEM

Download PPK

Score 30/30

Partition and Format a Disk Drive in Linux

1 hour 1 Credit Rate Lab

Introduction

In this lab, you'll learn how to partition and format a disk drive in Linux. Knowing how to do this is a critical skill to have as an IT Support Specialist. Partitions are important because a file system can't function without one. When you acquire a new disk drive, at least one partition is required in order to be able to write files to the file system. Different partitions can then have different file formats, depending on their purpose. For example, a disk partition that acts as a swap for your main memory may have a different file format than the default user-facing file systems. Partitions, like those used for system recovery, may also have different file formats. This shows you just how important this skill is to every IT Support Specialist out there.

Head's up: You'll experience a delay as the labs initially load (particularly for Windows labs). So, please **wait a couple of minutes for the labs to load**. Please also make sure to access the labs **directly through Coursera** and not in the Qwiklabs catalog. If you access the labs through the Qwiklabs catalog, you will **not** receive a grade. (As you know, a passing grade is required to matriculate through the course.) The grade is calculated when the lab is complete, so be sure to hit **End Lab** when you're done!

WARNING- If it's your second attempt of this lab, close this tab and **go back to Coursera** and retry this lab by hitting the "Open Tool button" in order to get a full score for this attempt.

You'll have 60 minutes to complete this lab.

What you'll do

You'll learn how to partition a disk drive into one or more partitions. You'll also learn how to format each of those partitions to a different file format. Your main learning objective for this lab is to practice the partitioning and formatting commands you'll find in this lab in the Linux VM.

Learning tip:

We encourage you to try and memorize all of these commands as best you can. With enough practice, using Linux commands will become second-nature to you. If you have access to your own Linux machine, try out the commands as you follow along in the next section.

in the next section.

If you don't have Linux available on your local machine, no worries! You can type these commands in a text editor, so you can refer back to them when you're doing the active lab exercises.

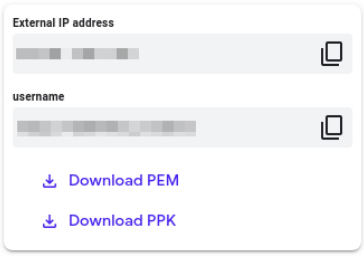
Start the lab

You'll need to start the lab before you can access the materials in the virtual machine OS. To do this, click the green "Start Lab" button at the top of the screen.

Note: For this lab you are going to access the **Linux VM** through your **local SSH Client**, and not use the **Google Console** (**Open GCP Console** button is not available for this lab).

Start Lab

After you click the "Start Lab" button, you will see all the SSH connection details on the left-hand side of your screen. You should have a screen that looks like this:



Accessing the virtual machine

Please find one of the three relevant options below based on your device's operating system.

Note: Working with Qwiklabs may be similar to the work you'd perform as an **IT Support Specialist**; you'll be interfacing with a cutting-edge technology that requires multiple steps to access, and perhaps healthy doses of patience and persistence(!). You'll also be using **SSH** to enter the labs -- a critical skill in IT Support that you'll be able to practice through the labs.

Option 1: Windows Users: Connecting to your VM

In this section, you will use the PuTTY Secure Shell (SSH) client and your VM's External IP address to connect.

Download your PPK key file

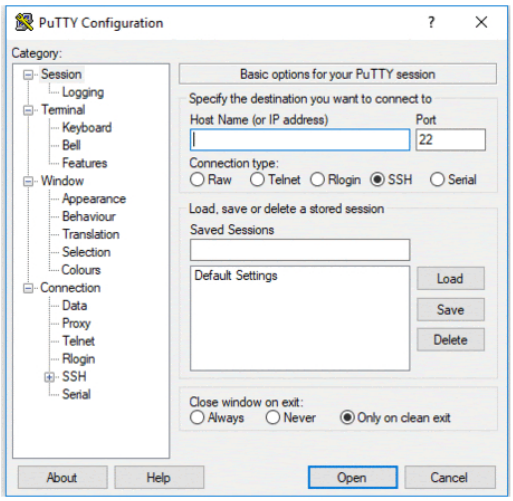
You can download the VM's private key file in the PuTTY-compatible **PPK** format from the Qwiklabs Start Lab page. Click on **Download PPK**.

Connect to your VM using SSH and PuTTY

1. You can download Putty from here

2. In the **Host Name (or IP address)** box, enter username@external_ip_address.

> **Note:** Replace **username** and **external_ip_address** with values provided in the lab.



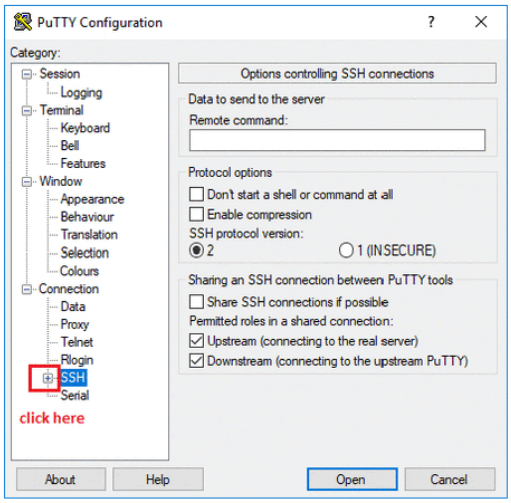
3. In the **Category** list, expand **SSH**.

4. Click **Auth** (don't expand it).

5. In the **Private key file for authentication** box, browse to the PPK file that you downloaded and double-click it.

6. Click on the **Open** button.

> **Note:** PPK file is to be imported into PuTTY tool using the Browse option available in it. It should not be opened directly but only to be used in PuTTY.



7. Click **Yes** when prompted to allow a first connection to this remote SSH server. Because you are using a key pair for authentication, you will not be prompted for a password.

Common issues

If PuTTY fails to connect to your Linux VM, verify that:

- You entered **<username>@<external ip address>** in PuTTY.
- You downloaded the fresh new PPK file for this lab from Qwiklabs.
- You are using the downloaded PPK file in PuTTY.

Option 2: OSX and Linux users: Connecting to your VM via SSH

Download your VM's private key file.

You can download the private key file in PEM format from the Qwiklabs Start Lab page. Click on **Download PEM**.

Connect to the VM using the local Terminal application

A **terminal** is a program which provides a **text-based interface for typing commands**. Here you will use your terminal as an SSH client to connect with lab provided Linux VM.

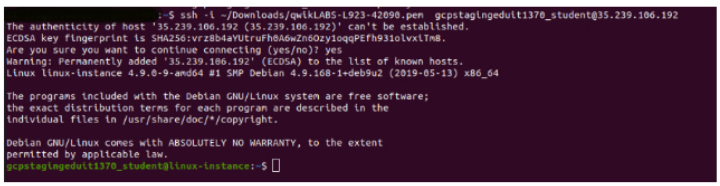
1. Open the Terminal application.
   - To open the terminal in Linux use the shortcut key **Ctrl+Alt+t**.
   - To open terminal in **Mac** (OSX) enter **cmd + space** and search for **terminal**.
2. Enter the following commands.

> **Note:** Substitute the **path/filename for the PEM** file you downloaded, **username** and **External IP Address**.

You will most likely find the PEM file in **Downloads**. If you have not changed the download settings of your system, then the path of the PEM key will be **~/Downloads/qwikLABS-XXXXX.pem**

```
chmod 600 ~/Downloads/qwikLABS-XXXXX.pem
```

```
ssh -i ~/Downloads/qwikLABS-XXXXX.pem username@External Ip Address
```



Option 3: Chrome OS users: Connecting to your VM via SSH

> **Note:** Make sure you are not in **Incognito/Private mode** while launching the application.

Download your VM's private key file.

You can download the private key file in PEM format from the Qwiklabs Start Lab

You can download the private key file in PEM format from the QwikLabs Start Lab page. Click on **Download PEM**.

Connect to your VM

1. Add Secure Shell from here to your Chrome browser.
2. Open the Secure Shell app and click on **[New Connection]**.

3. In the **username** section, enter the username given in the Connection Details Panel of the lab. And for the **hostname** section, enter the external IP of your VM instance that is mentioned in the Connection Details Panel of the lab.

4. In the **Identity** section, import the downloaded PEM key by clicking on the **Import...** button beside the field. Choose your PEM key and click on the **OPEN** button.

> **Note:** If the key is still not available after importing it, refresh the application, and select it from the **Identity** drop-down menu.

5. Once your key is uploaded, click on the **[ENTER] Connect** button below.

6. For any prompts, type **yes** to continue.
7. You have now successfully connected to your Linux VM.

You're now ready to continue with the lab!

Blocks and partitions

Before diving into the details of creating partitions and formatting them, let's kick things off with a review of blocks and partitions.

Blocks

Blocks are a layer of storage devices that allow individual access to each independently. They allow programs to access storage without worrying about whether the underlying hardware device is a hard drive, solid state drive, flash drive, etc.

In Linux, you can view block devices and file systems attached to your system using the **lsblk** command. This command gathers information about all devices attached to the system, and prints them out using a tree-like structure. To view the devices attached to your VM, use the **lsblk** command.

```
lsblk
```

```
eduit914728_student@linux-instance:~$ lsblk
NAME   MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda      8:0    0  10G  0 disk
└─sda1   8:1    0  10G  0 part
sdb      8:16   0  10G  0 disk
└─sdb1   8:17   0  10G  0 part /
```

You'll see that your instance has two block devices attached to it (disks). Each of them is 10GB in size. The column MOUNTPOINT shows where a block device is mounted. It's from this location that files on the disk can be accessed. In this case, the MOUNTPOINT is displaying "**/**" against **sdb**, which means the second disk (sdb) is mounted at the root of the Linux file system tree. Thus, the files you're seeing on your system right now are from this disk.

A first disk, **sda**, is also available, but it's not mounted. In this lab, you'll divide this disk into two partitions. You'll then mount one of these partitions onto the file system, so you can start accessing files from it.

> **Note**: These may be swapped for you, and your VM may be mounted on sda instead of sdb. This will change the commands used in the lab, so when you see \[MOUNT DRIVE\] replace it with your mount drive (sda or sdb) and when you see \[SECOND DRIVE\] replace it with the other one. If your VM is mounted on sda, the screenshots will also be flipped from what you will see.

Optionally, you can view disks mounted on the system using the **df** command. This command is normally used to display the amount of space available on the file system. It lists all block devices with the available space on them. Use the **-h** option to display file sizes in human readable format.

```
df -h
```

```
eduit914728_student@linux-instance:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.8G     0  1.8G   0% /dev
tmpfs           371M  6.4M  365M   2% /run
/dev/sdb1       9.8G  1.1G  8.3G  12% /
tmpfs           1.9G     0  1.9G   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           1.9G     0  1.9G   0% /sys/fs/cgroup
```

Partitions

Instead of using a storage block as a whole, it's common practice to divide a storage block into different partitions. Partitions can be different sizes, and formatted to different filesystems. This allows you to use a single storage device for different purposes.

You can display partition information using the **fdisk** command. You can also use the -l option to list partitions in the block. You can pass a device name to the fdisk command to list the partitions contained in that device.

To list all partitions, use **fdisk -l**

```
sudo fdisk -l
```

```
edult914728_student@linux-instance:~$ sudo fdisk -l
Disk /dev/sdb: 10 GiB, 10737418240 bytes, 20971520 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: dos
Disk identifier: 0xc2b76e68

Device     Boot Start      End  Sectors Size Id Type
/dev/sdb1  *     4096 20971519 20967424  10G 83 Linux


Disk /dev/sda: 10 GiB, 10737418240 bytes, 20971520 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: dos
Disk identifier: 0xc2b76e68

Device     Boot Start      End  Sectors Size Id Type
/dev/sda1  *     4096 20971519 20967424  10G 83 Linux
edult914728_student@linux-instance:~$
```

To list partitions contained in **/dev/sdb**, pass **/dev/sdb** to the **fdisk** command.

```
sudo fdisk -l /dev/sdb
```

```
edult914728_student@linux-instance:~$ sudo fdisk -l /dev/sdb
Disk /dev/sdb: 10 GiB, 10737418240 bytes, 20971520 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: dos
Disk identifier: 0xc2b76e68

Device     Boot Start      End  Sectors Size Id Type
/dev/sdb1  *     4096 20971519 20967424  10G 83 Linux
edult914728_student@linux-instance:~$
```

fdisk displays information contained in the partition table, where information about partitions is stored.

Disk partitioning with fdisk

When the *fdisk* command is used without options, it provides a menu-driven environment for creating and deleting partitions.

> **Caution!**: Modifying partitions is destructive, and can lead to loss of data. Not good! Remember to always backup your data before modifying partitions on a live system.

Mount and umount

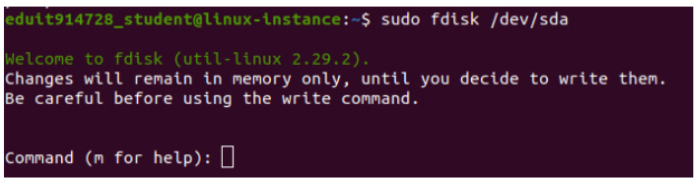
Mounting and unmounting mean making devices available or unavailable on a Linux file system. This is accomplished by the commands *mount* and *umount*. Before modifying a disk, you should first **unmount** it from the system, using the umount command. When modifications on the disk are done, you should **mount** it back onto the system. For this exercise, since the device we're partitioning isn't initially mounted, you can proceed with partitioning.

Go ahead and start *fdisk* in interactive mode by passing the name of the disk you want to partition. In this lab, we'll partition **/dev/sda**

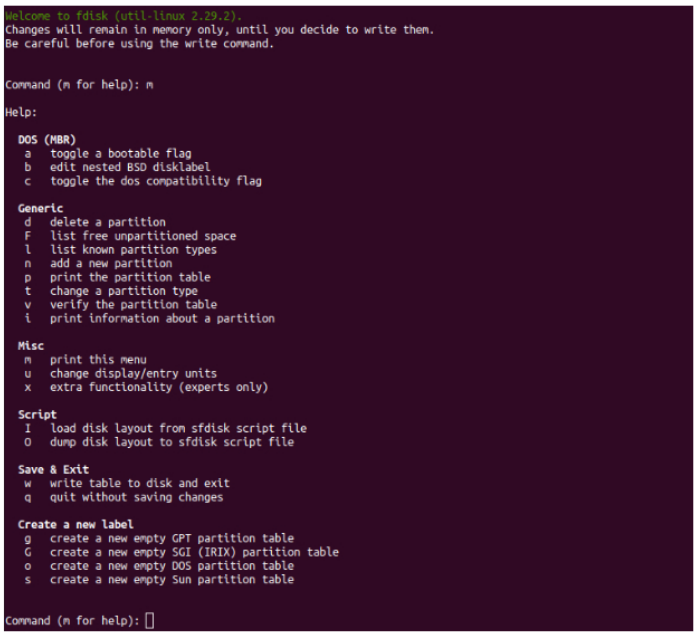
> **Note**: We will partition the disk that's not currently mounted. You should select dev/sdb if /dev/sda is where the operating system is mounted, and /dev/sda otherwise. You can still partition the disk even when the operating system is running from it, but a reboot will be required in order for the partition changes you make to take place.

Start fdisk by passing the disk you want to partition as the parameter.

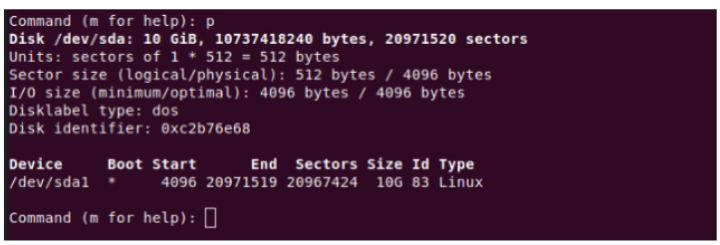
```
sudo fdisk /dev/[SECOND DRIVE]
```



fdisk will start in interactive mode. You can use **m** to use help provided by the command.



You can use **p** to show details about partitions on the disk.



Enter **q** to exit interactive mode when you are finished exploring.

Creating Partitions

You'll now create new partitions using **fdisk**. You'll partition **the second drive** into two partitions: one swap partition of size **1GB**, and another of size **9GB**. The file system type on the second partition will be ext4.

Open *fdisk* in interactive mode to do the partitioning:

```
sudo fdisk /dev/[SECOND DRIVE]
```

```
eduit914728_student@linux-instance:~$ sudo fdisk /dev/sda

Welcome to fdisk (util-linux 2.29.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help):
```

To create a new partition, the command control **n** is used. However, since all the space on the disk is currently allocated, you'll need to first free up space by deleting the default partition.

Use the **d** command control to delete the default partition. When you issue the **d** command control, **fdisk** asks you to enter the number of partitions you want to delete. Since you only have one partition, the default one, **fdisk** will automatically select and delete it to continue.

```
Command (m for help): d
Selected partition 1
Partition 1 has been deleted.
```

You're now able to create your new partitions. Enter the command control for creating a new partition, **n**.

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p):
```

fdisk will present you with two options to select from: **p** for primary, and **e** for extended or logical partition. Since we want to create the partitions on the actual physical disk, select **p** by pressing **Enter**.

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p):

Using default response p.
Partition number (1-4, default 1):
```

Next, you'll need to provide the partition number for the new partition. Since it's a primary partition, it can only be labelled from 1-4. It's good practice to assign partition numbers sequentially; problems can sometimes arise with certain

programs if partitions aren't ordered sequentially. Give the number **1** to this first partition by pressing **Enter**, or optionally entering **1**.

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p):

Using default response p.
Partition number (1-4, default 1):
First sector (2048-20971519, default 2048):
```

You'll then need to provide the starting sector (memory location) of the new partition, from where you want to allocate. Here, press **Enter** to select the default value 2048.

```
Select (default p):

Using default response p.
Partition number (1-4, default 1):
First sector (2048-20971519, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-20971519, default 20971519):
```

Provide the last sector of the new partition, up to where you want to allocate. The difference between the first and last sectors makes up the total size of the partition. Disk sector represents units used to measure the size on disks. Each sector stores a fixed amount of data. In lots of hard disks, for example, a sector stores 512 bytes. To create the first 1GB partition, enter **2097200** (divide the original partition by 10).

```
Using default response p.
Partition number (1-4, default 1):
First sector (2048-20971519, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-20971519, default 20971519): 2097200

Created a new partition 1 of type 'Linux' and of size 1023 MiB.

Command (m for help):
```

Two important things happen here: the partition size is set to **1GB**, and the partition type is set to **Linux**. (You'll see how to change partition types in the next section.) Voila! One partition is now created. You'll now move on to the second one.

Use the command control **n** again for a new partition.

```
Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p):
```

Select **p** for a primary partition.

```
Select (default p):

Using default response p.
Partition number (2-4, default 2):
```

Select partition number **2** to issue partition numbers in sequence.

```
Select (default p):

Using default response p.
Partition number (2-4, default 2):
First sector (2097201-20971519, default 2099200):
```

Select the default partition starting sector, which is the next sector from the last partition you allocated.

```
Using default response p.
Partition number (2-4, default 2):
```

```
First sector (2097201-20971519, default 2099200):
Last sector, +sectors or +size{K,M,G,T,P} (2099200-20971519, default 20971519):
```

Also select the default last sector, which will be the last sector of the remaining disk space.

```
Using default response p.
Partition number (2-4, default 2):
First sector (2097201-20971519, default 2099200):
Last sector, +sectors or +size{K,M,G,T,P} (2099200-20971519, default 20971519):

Created a new partition 2 of type 'Linux' and of size 9 GiB.

Command (m for help):
```

The second partition is now created. Sweet!

Before committing your changes, you'll change the second partition to a different partition type. You'll change the first partition type to a Linux swap type. Enter command control **t** to change the partition type, and select the first partition.

```
Command (m for help): t
Partition number (1,2, default 2): 1
Partition type (type L to list all types):
```

You can use the command control **L** to view a list of all partition types.

```
Command (m for help): t
Partition number (1,2, default 2): 1
Partition type (type L to list all types): L

 0  Empty           24  NEC DOS         81  Minix / old Lin bf  Solaris
 1  FAT12           27  Hidden NTFS Win 82  Linux swap / So c1  DRDOS/sec (FAT-
 2  XENIX root      39  Plan 9          83  Linux           c4  DRDOS/sec (FAT-
 3  XENIX usr       3c  PartitionMagic  84  OS/2 hidden or  c6  DRDOS/sec (FAT-
 4  FAT16 <32M      40  Venix 80286     85  Linux extended  c7  Syrinx
 5  Extended        41  PPC PReP Boot   86  NTFS volume set da  Non-FS data
 6  FAT16           42  SFS             87  NTFS volume set db  CP/M / CTOS / .
 7  HPFS/NTFS/exFAT 4d  QNX4.x          88  Linux plaintext de  Dell Utility
 8  AIX             4e  QNX4.x 2nd part 8e  Linux LVM       df  BootIt
 9  AIX bootable    4f  QNX4.x 3rd part 93  Amoeba          e1  DOS access
 a  OS/2 Boot Manag 50  OnTrack DM      94  Amoeba BBT      e3  DOS R/O
 b  W95 FAT32       51  OnTrack DM6 Aux 9f  BSD/OS          e4  SpeedStor
 c  W95 FAT32 (LBA) 52  CP/M            a0  IBM Thinkpad hi ea  Rufus alignment
 e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a5  FreeBSD         eb  BeOS fs
 f  W95 Ext'd (LBA) 54  OnTrackDM6      a6  OpenBSD         ee  GPT
10  OPUS            55  EZ-Drive        a7  NeXTSTEP        ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a8  Darwin UFS      f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a9  NetBSD          f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       ab  Darwin boot     f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys af  HFS / HFS+      f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fb  VMware VMFS
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fc  VMware VMKCORE
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid fd  Linux raid auto
1c  Hidden W95 FAT3 75  PC/IX           bc  Acronis FAT32 L fe  LANstep
1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot    ff  BBT
Partition type (type L to list all types):
```

Enter **82** to change the partition type to 'Linux swap / Solaris', and press **Enter**. Head's up: Some of the characters in the partition type name **Linux swap / Solaris** are truncated.

```
Partition type (type L to list all types): 82

Changed type of partition 'Linux' to 'Linux swap / Solaris'.

Command (m for help):
```

The partition type will be changed to match the selection.

Up to this point, you've just been editing the partition table in memory. You can use the **q** command here to quit **fdisk** without committing changes to the disk. You can also update your partitions by using the **d** and **n** commands to remove and add new partitions.

You can also use the **v** command here to verify your changes before proceeding.

```
Command (m for help): v
Remaining 1999 unallocated 512-byte sectors.

Command (m for help):
```

If you're satisfied with the changes you've made so far, you can commit them to the disk by using the **w** command.

disk by using the **w** command.

```
Command (m for help): w

The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

eduit914728_student@linux-instance:~$
```

Congrats! You've successfully partitioned the second disk using **fdisk**.

The second disk device is now made up of two partitions of **1GB** and **9GB**, respectively.

Click *Check my progress* to verify the objective.

Partitioning

Check my progress

Formatting partitions using mkfs

Next, you'll create different file systems in the partitions you just created. You'll do this by using the command **mkfs** in Linux. Multiple filesystem types exist, and it's important to know all of them, along with the functions they're best suited for. In this lab, you'll format the second partition into ext4, the most widely used Linux filesystem type.

To do this, use **lsblk** again to find the disk you want to create the file system type in.

```
lsblk
```

```
eduit914728_student@linux-instance:~$ lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   10G  0 disk
├─sda1   8:1    0 1023M  0 part
└─sda2   8:2    0    9G  0 part
sdb      8:16   0   10G  0 disk
└─sdb1   8:17   0   10G  0 part /
```

Format the second partition **in your unmounted drive** (sdb2 or sda2) to ext4 using this command:

```
sudo mkfs -t ext4 /dev/[SECOND DRIVE]2
```

```
eduit914728_student@linux-instance:~$ sudo mkfs -t ext4 /dev/sda2
mke2fs 1.43.4 (31-Jan-2017)
Discarding device blocks: done
Creating filesystem with 2359040 4k blocks and 589824 inodes
Filesystem UUID: 47578d10-9174-4450-9d7d-402b344e0cbb
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

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

Click *Check my progress* to verify the objective.

EXT4

Check my progress

You can now mount **/dev/sda2** to a location on the file system to start accessing files on it. Mount it on the directory **/home/my_drive**.

```
sudo mount /dev/[SECOND DRIVE]2 /home/my_drive
```

```
:~$ sudo mount /dev/sda2 /home/my_drive
:~$
```

From now on, accessing "/home/my_drive" will be accessing files on the disk.

That's it! You've successfully partitioned and formatted a disk in Linux.

Click *Check my progress* to verify the objective.

Mount

Check my progress

Conclusion

In this lab, we've gone through the process of creating partitions, formatting them to specific filesystems, and mounting them onto accessible locations in Linux. You should continue to practice these commands so that you become comfortable using them.

End your lab

When you have completed your lab, click **End Lab**. Qwiklabs removes the resources you've used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.

The number of stars indicates the following:

- 1 star = Very dissatisfied
- 2 stars = Dissatisfied
- 3 stars = Neutral
- 4 stars = Satisfied
- 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.