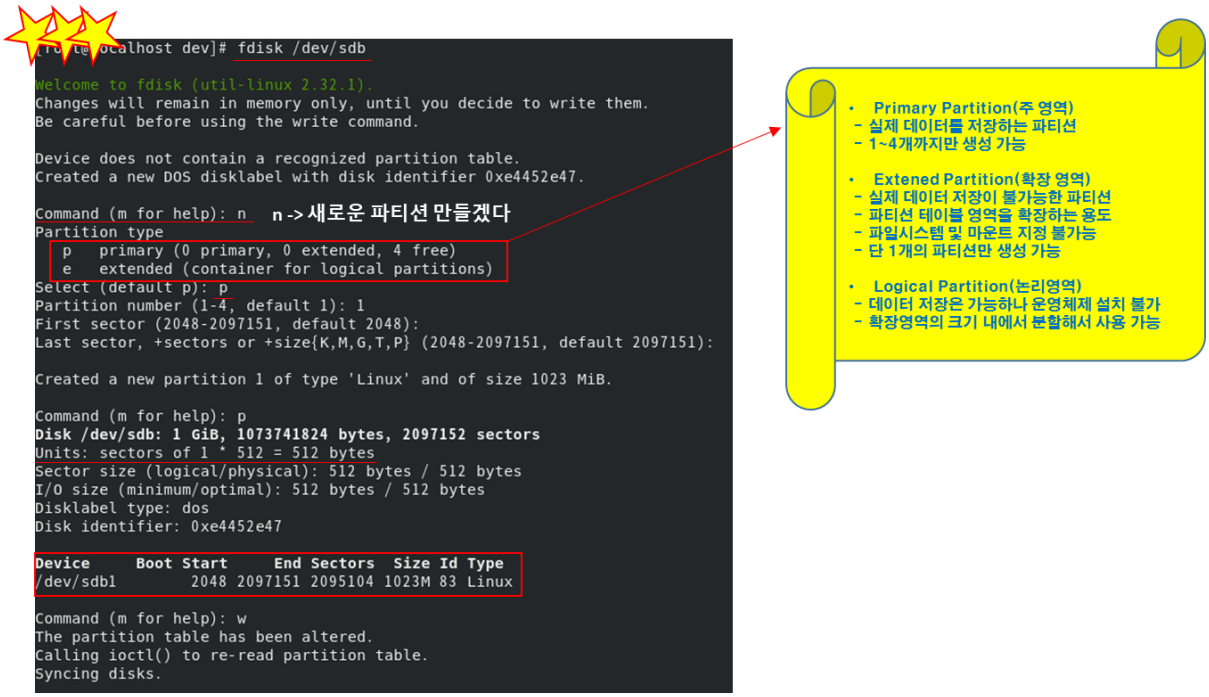
[4일차]



fdisk 명령어는 파티션을 나눠주는 역할을 수행하는 명령어이다.

첫 실행 후 n → 새로운 파티션을 생성하는 것

Partiotion type에는 Primary, Extend 두 가지의 선택이 가능한데, 자세한 설명은 위의 노란색 박스를 읽어보자. 일반적으로 p를 선택하여 주 영역 파티션으로 파티션을 배분한다.

파티션의 섹터는 1섹터당 512byte 의 용량을 가지는데, 현재 그림에 나와있듯이 2,097,152 개의 파티션을 계산해보면 다음의 식과 같이 계산되어 나온다.

1G = 1024M = 1024 * 1024 (1,048,576) Kbyte = 2,097,152 Sector

fdisk를 사용하여 파티션을 나누었다면, 꼭 p를 눌러서 현재 파티션 상태를 확인한 후 w로 저장하자!

```
[root@localhost dev]# mkfs.ext4 /dev/sdb1
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 261888 4k blocks and 65536 inodes
Filesystem UUID: 0d4cd113-5342-4387-86ae-7757c83aac79
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: done
```

mkfs 명령어는 파티션을 원하는 파일시스템으로 포멧하는 명령어이다.

명령어는 두 가지 방식으로 표현이 가능하지만, mkfs.파일시스템 장치위치 와 같은 형태로 사용하는 것이 편리하다.

fdisk를 사용한 파티션 분할과 mkfs로 디스크 포멧까지 완료하였다면, 마운트를 통해 사용 가능하도록 등록해주어야 한다. 마운트를 위하여 다음과 같이 디렉터리를 생성한 후 마운트를 수행하자.

mount /dev’sdb1 /mydata

```
[root@localhost dev]# mkdir /mydata
[root@localhost dev]#
[root@localhost dev]# cd /
[root@localhost /]# ls
bin   dev  home  lib64  mnt     opt   root  sbin  sys  usr
boot  etc  lib   media  mydata  proc  run   srv   tmp  var
[root@localhost /]#
```

```
[root@localhost ~]# ls
anaconda-ks.cfg       공개      문서      비디오  서식
initial-setup-ks.cfg  다운로드  바탕화면  사진    음악
[root@localhost ~]# cp anaconda-ks.cfg /mydata/test1
[root@localhost ~]#
[root@localhost ~]# ls -l /mydata
합계 4
-rw------- 1 root root 1591  8월  9 00:34 test1
[root@localhost ~]#
```

마운트를 완료하였으면, 재부팅 시 자동으로 마운트 되어있을 수 있도록 설정하는 것이 필요로 하다.

이는 /etc/fstab 파일에 설정을 입력해주는 것으로 해결할 수 있다.

```
[root@localhost /]# vi /etc/fstab
```

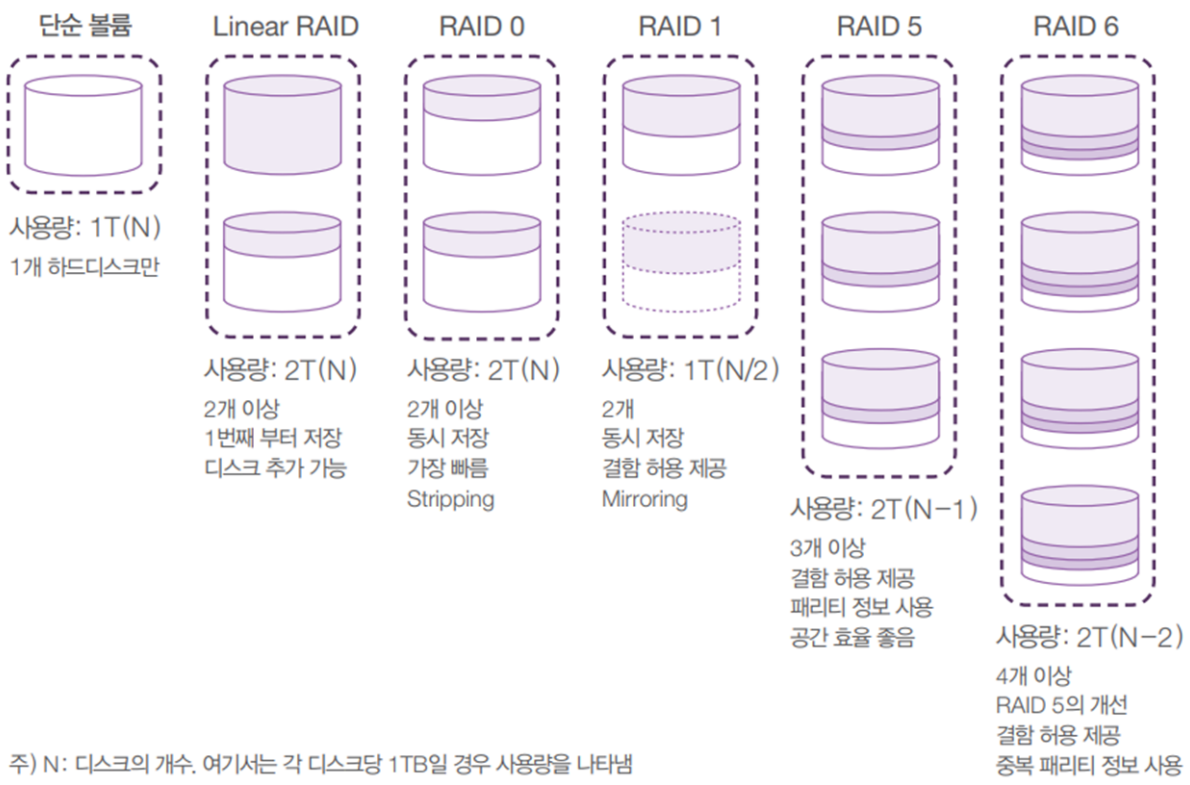
```
# /etc/fstab
# Created by anaconda on Sun Aug  6 21:04:23 2023
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
UUID=fd92c667-c396-4445-8e26-26acf6af6e6a /                       xfs     defaults        0 0
UUID=4a7651c3-f9e1-4abb-aa13-d91bb399c595 swap                    swap    defaults        0 0

/dev/sdb1       /mydata         ext4            defaults        0       0
```

/etc/fstab을 사용하는 방법은 아래와 같다.

(장치명) (마운트할 디렉터리) (파일시스템) (사용방법) (덤프 여부) (디스크 체크여부)

여기에서 사용방법은 일반적으로 defaults, 덤프와 디스크 체크여부는 0으로 사용한다.

RAID



RAID는 여러개의 디스크를 하나의 디스크처럼 사용하는 방식으로, 하드웨어와 소프트웨어로 구분함

Linear RAID

- 최소 2개의 하드디스크 필요
- 2개 이상의 하드디스크를 통합하여 1개의 큰 디스크처럼 사용함
- 디스크 저장 시 앞 디스크부터 차례대로 저장함
- 한 개의 디스크가 고장나는 경우 데이터를 복구할 수 없음

RAID 0

- 최소 2개의 하드디스크 필요
- Linear RAID와 동일한 효과를 가지지만, 디스크 저장 시 분할하여 저장함

RAID 1

- 최소 2개의 하드디스크 필요
- 하나의 디스크에 작성하는 내용을 다른 하나의 디스크에 동일하게 작성함
- 이를 Mirorring이라고 하며, 한 개의 디스크가 고장나도 다른 하나의 디스크가 역할을 수행함
- 데이터를 저장하는 효율성은 떨어지지만, 신뢰성이 높고 중요한 데이터를 저장하기에 적합함

RAID 5

- 최소 3개의 하드디스크 필요
- Parity bit를 사용하여 장애가 발생했을 때, 장애난 디스크의 정보를 유추하여 복구함
- 데이터 사용 가능 용량은 2/3만큼의 용량을 사용 가능함
- RAID 1의 데이터 안정성과 RAID 0의 공간 효율성을 둘다 챙길 수 있음

RAID 10

- 최소 4개의 디스크가 필요함
- 두개의 디스크를 RAID0으로 묶어 용량을 늘리고, 각 pair를 또 묶어서 복제본을 만듬
- 신뢰성과 성능을 모두 잡을 수 있지만, 디스크의 낭비가 발생함

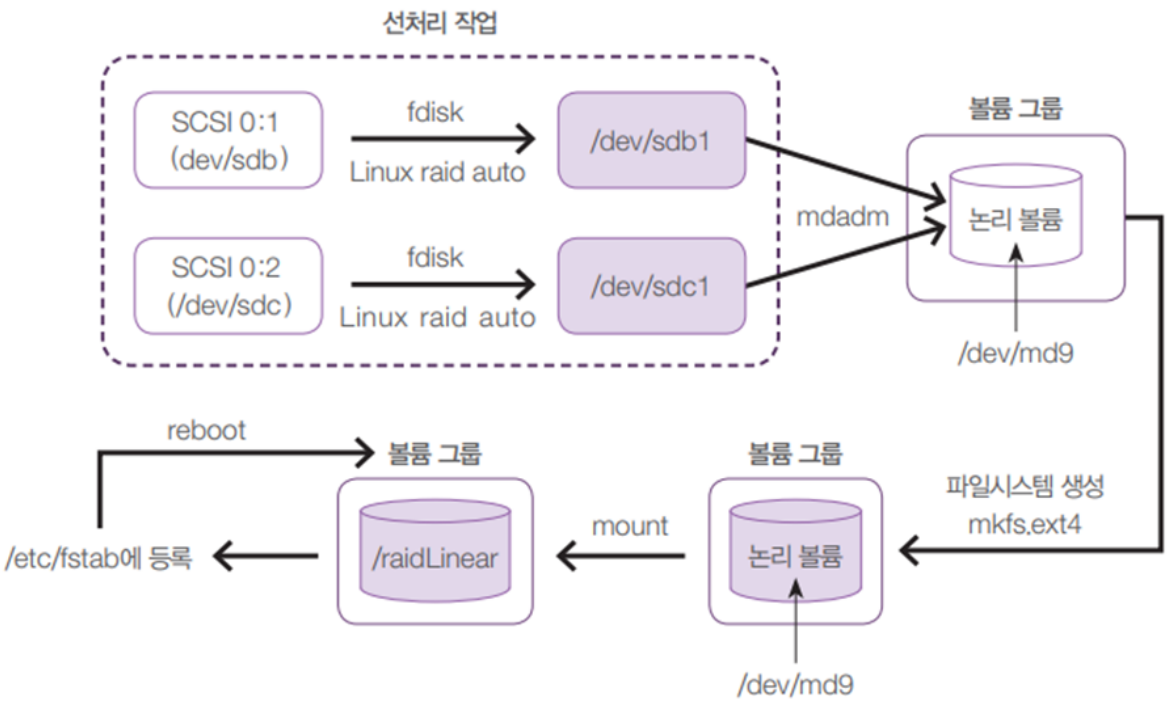
RAID 6

- RAID 6은 RAID 5의 디스크가 두 개 이상 장애발생 시 복구를 할 수 없는 문제를 해결하기 위함
- 공간 효율성은 RAID 5에 비해 떨어지나 신뢰성은 높은 형태

```
Welcome to fdisk (util-linux 2.32.1).
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 0xe5ee98f8.

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 (2048-4194303, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-4194303, default 4194303):

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

Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): fd
Changed type of partition 'Linux' to 'Linux raid autodetect'.

Command (m for help): p
Disk /dev/sdb: 2 GiB, 2147483648 bytes, 4194304 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
Disklabel type: dos
Disk identifier: 0xe5ee98f8

Device     Boot Start     End Sectors Size Id Type
/dev/sdb1        2048 4194303 4192256   2G fd Linux raid autodetect

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

RAID 기능을 사용할 때, 파티션을 분할하고 파티션에 fd(Linux raid autodetect)옵션을 주어 사용함

이외에는 기존 파티션 생성 방법과 동일하다.




```
[root@localhost ~]# mdadm --create /dev/md9 --level=linear --raid-device=2 /dev/
sdb1 /dev/sdc1
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md9 started.
[root@localhost ~]# mdadm --detail --scan
ARRAY /dev/md9 metadata=1.2 name=localhost.localdomain:9 UUID=2865a464:3d20aa32:
b03ad162:f1580b81
```

mdadm 설명

- --create /dev/md9 → md9 장치에 RAID생성
- --level=linear → Linear RAID로 지정

- --raid-devices=2 /dev/sdb1 /dev/sdc1
  → raid 사용할 장치는 2개인데, sdb1과 sdc1을 묶어달라

- mdadm --stop /dev/md9 → RAID 장치인 /dev/md9 중지
- mdadm --run /dev/md9 → 중지된 RAID장치 기동
- mdadm --detail /dev/md9 → 상세 내역 출력

RAID는 mdadm 명령어를 사용하여 생성 할 수 있으며 생성 이후의 작업은 일반적인 하드디스크 추가하는 내용과 동일하다.

RAID의 방식에 따라서 —level= 부분에 linear, 0, 1, 5, 6, 10과 같이 입력하면 해당 레이드의 디스크가 생성된다.

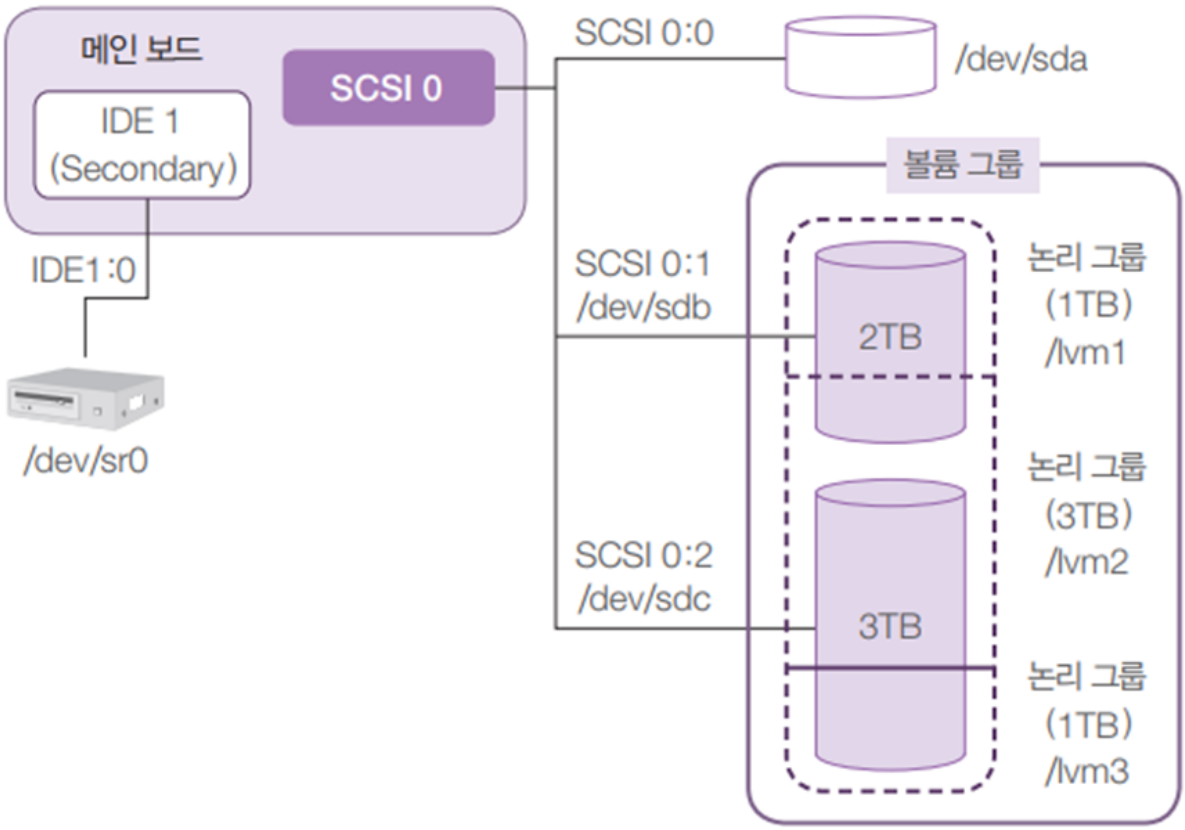
만약 하드디스크에 장애가 발생하는 경우 다음과 같이 처리한다.

```
[root@localhost ~]# mdadm --stop /dev/md9
mdadm: stopped /dev/md9
[root@localhost ~]#
[root@localhost ~]# mdadm --create /dev/md9 --level=linear --raid-device=2 /dev/sdb1 /dev/sdc1
mdadm: /dev/sdb1 appears to be part of a raid array:
       level=linear devices=2 ctime=Wed Aug  9 01:36:05 2023
Continue creating array? y
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md9 started.
[root@localhost ~]#
[root@localhost ~]# mdadm --stop /dev/md0
mdadm: stopped /dev/md0
[root@localhost ~]#
[root@localhost ~]# mdadm --create /dev/md0 --level=0 --raid-device=2 /dev/sdd1 /dev/sde1
mdadm: /dev/sdd1 appears to be part of a raid array:
       level=raid0 devices=2 ctime=Wed Aug  9 01:50:55 2023
Continue creating array? y
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# mdadm --stop /dev/md1
mdadm: Cannot get exclusive access to /dev/md1:Perhaps a running process, mounted filesystem or active volume group?
[root@localhost ~]#
[root@localhost ~]# mdadm /dev/md1 --add /dev/sdg1
mdadm: added /dev/sdg1
[root@localhost ~]#
[root@localhost ~]# mdadm /dev/md5 --add /dev/sdi1
mdadm: added /dev/sdi1
```

Linear와 RAID 0의 경우에는 장애가 발생한 레이드를 종료시키고 다시 mdadm을 사용하여 레이드용 디스크를 생성하고 추가한다.

이외의 경우에는 mdadm (RAID 장치) —add (추가디스크) 의 명령어로 디스크를 추가해주면 된다.

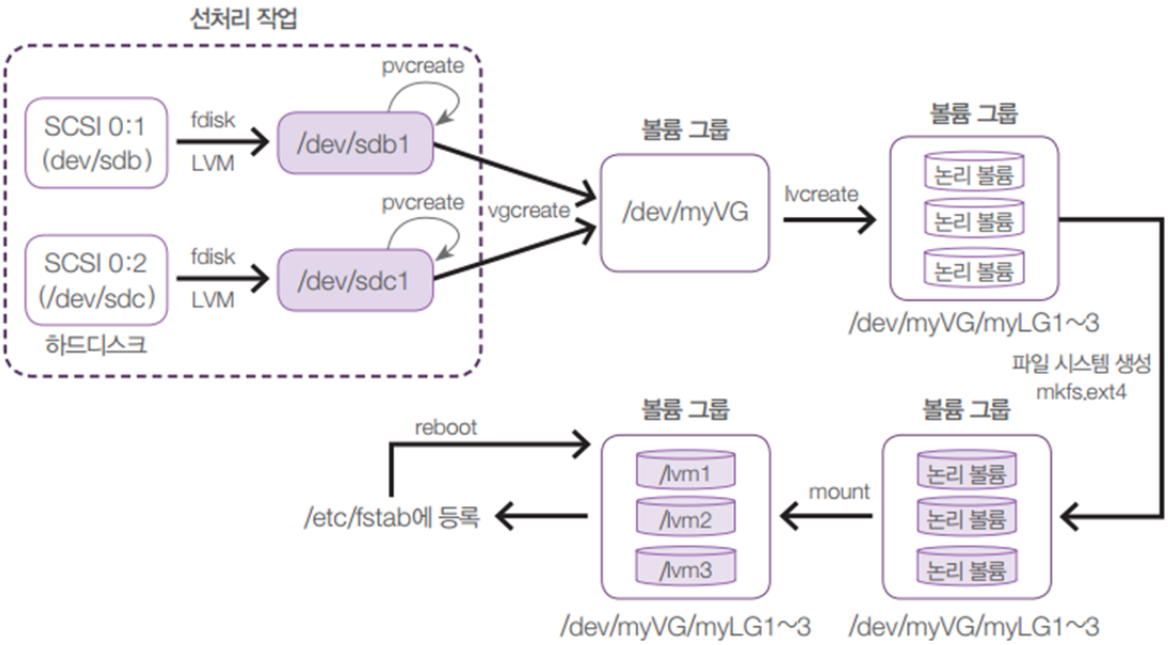
LVM



LVM은 디스크를 통합한 뒤 논리적인 크기로 분할하는 기법을 말한다.

LVM의 구성은 세 가지 구성으로 구분할 수 있다.

Physical Volume(물리 볼륨) : /dev/sda1, /dev/sdb1 등의 파티션
Volume Group(볼륨 그룹) : 물리 볼륨을 합쳐서 1개의 물리 그룹으로 만드는 것
Logical Volume(논리 볼륨) : 볼륨 그룹을 1개 이상으로 나눠서 논리 그룹으로 나눈 것



우선 fdisk를 lvm용 파티션을 생성하고, 물리 볼륨을 생성한다.

이후 생성한 물리 볼륨들을 하나의 그룹으로 만들어 볼륨 그룹을 만들고, 해당 그룹을 원하는 크기에 맞추어 논리 볼륨으로 생성한다. 이후 각 볼륨을 포멧하고 마운트하여 사용한다.

```
[root@localhost ~]# fdisk /dev/sdb

Welcome to fdisk (util-linux 2.32.1).
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 0xb6a2856b.

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 (2048-6291455, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-6291455, default 6291455):

Created a new partition 1 of type 'Linux' and of size 3 GiB.

Command (m for help): t
Selected partition 1
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: 3 GiB, 3221225472 bytes, 6291456 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
Disklabel type: dos
Disk identifier: 0xb6a2856b

Device     Boot Start     End Sectors Size Id Type
/dev/sdb1        2048 6291455 6289408   3G 8e Linux LVM

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

lvm용 파티션을 생성할때 옵션은 8e옵션을 주어 생성하고, p로 확인해보면 LVM 으로 생성된것을 확인 가능하다. 이때 LVM이 아니라면 이후 물리볼륨을 생성할 수 없다.

```
[root@localhost ~]# pvcreate /dev/sdb1
  Physical volume "/dev/sdb1" successfully created.
[root@localhost ~]# pvcreate /dev/sdc1
  Physical volume "/dev/sdc1" successfully created.
[root@localhost ~]#
[root@localhost ~]# vgcreate myVG /dev/sdb1 /dev/sdc1
  Volume group "myVG" successfully created
[root@localhost ~]#
[root@localhost ~]# vgdisplay
  --- Volume group ---
  VG Name               myVG
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                2
  Act PV                2
  VG Size               4.99 GiB
  PE Size               4.00 MiB
  Total PE              1278
  Alloc PE / Size       0 / 0
  Free  PE / Size       1278 / 4.99 GiB
  VG UUID               pgWIct-0fee-qloK-2EgJ-pHBL-eQbZ-2Yopsz
```

```
[root@localhost ~]# lvcreate --size 1G --name myLG1 myVG
  Logical volume "myLG1" created.
[root@localhost ~]# lvcreate --size 3G --name myLG2 myVG
  Logical volume "myLG2" created.
[root@localhost ~]# lvcreate --extents 100%FREE --name myLG3 myVG
  Logical volume "myLG3" created.
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# ls -l /dev/myVG
합계 0
lrwxrwxrwx 1 root root 7  8월  9 22:35 myLG1 -> ../dm-0
lrwxrwxrwx 1 root root 7  8월  9 22:35 myLG2 -> ../dm-1
lrwxrwxrwx 1 root root 7  8월  9 22:37 myLG3 -> ../dm-2
[root@localhost ~]#
```

```
[root@localhost ~]# mkfs.ext4 /dev/myVG/myLG1
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 262144 4k blocks and 65536 inodes
Filesystem UUID: 8ae3c55c-bb1f-4cc9-a1ef-709cf58a6b25
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

[root@localhost ~]# mkfs.ext4 /dev/myVG/myLG2
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 786432 4k blocks and 196608 inodes
Filesystem UUID: e9940ab7-ed8c-476c-8a1e-1b981e2b5a2f
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@localhost ~]# mkfs.ext4 /dev/myVG/myLG3
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 260096 4k blocks and 65024 inodes
Filesystem UUID: 8ba93b61-b9b6-4c39-99d0-ff1cb397d182
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: done
```

```
[root@localhost ~]# mkdir /lvm1
[root@localhost ~]# mkdir /lvm2
[root@localhost ~]# mkdir /lvm3
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# mount /dev/myVG/myLG1 /lvm1
[root@localhost ~]# mount /dev/myVG/myLG2 /lvm2
[root@localhost ~]# mount /dev/myVG/myLG3 /lvm3
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# df -h
Filesystem              Size  Used Avail Use% Mounted on
devtmpfs                637M     0  637M   0% /dev
tmpfs                   652M     0  652M   0% /dev/shm
tmpfs                   652M  9.6M  642M   2% /run
tmpfs                   652M     0  652M   0% /sys/fs/cgroup
/dev/sda2                76G  5.5G   71G   8% /
tmpfs                   131M   28K  131M   1% /run/user/42
tmpfs                   131M  4.6M  126M   4% /run/user/0
/dev/mapper/myVG-myLG1  976M  2.6M  907M   1% /lvm1
/dev/mapper/myVG-myLG2  2.9G  9.0M  2.8G   1% /lvm2
/dev/mapper/myVG-myLG3  985M  2.5M  915M   1% /lvm3
```

수업 중 alias c=clear 가 동작하지 않았던 이슈에 대해 확인해보았는데,

단일 문자로 설정할 경우 동작하지 않는것 같네요.

alias cl=clear 로 설정하면 정상적으로 동작합니다.

단, 명령어에 띄어쓰기가 있는 경우 아래와 같이 작성하여 단축어로 사용 가능합니다

alias lss='ls -als'

7장 셸스크립트의 내용은 제공한 자료를 참조하길 바랍니다.