- Ch_1 _ RAID 원상 복구
- Ch_2 _ RAID6, 10 (1+0) 구축
- Ch_3 _ RAID6, 10 문제 발생 테스트
- Ch_4 _ LVM 개념과 실습
- Ch_5 _ 쿼터 개념과 실습
- Ch_6 _ 셸의 기본과 작성법

- RAID 정상 복구
  - 고장난 디스크 4 개를 새 디스크로 교체
  - edit virtual machien setting > 하드디스크 4 개 추가

```
ubuntu@server:~$ sudo mdadm /dev/md1 --add /dev/sdg1
mdadm: added /dev/sdg1
```

➔ RAID1 같은 경우는 '결함 허용' 이기 때문에 잘 실행되는 디스크에 추가만 해줌

- --add 사용 (md1)

```
ARRAY /dev/md1 metadata=1.2 name=se
ARRAY /dev/md/server:0 metadata=1.2
ARRAY /dev/md/server:9 metadata=1.2
ARRAY /dev/md/server:5 metadata=1.2
```

➔ 재 부팅 후 이름이 바뀌어 있는 상황

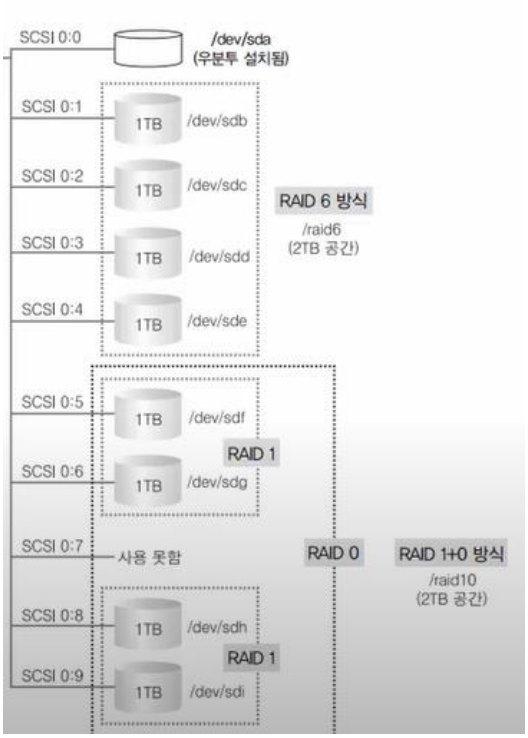
- 종종 있는 현상으로, /etc/fstab 에 변경된 이름으로 다시 저장하면 해결됨

```
/dev/loop11         44800   44800         0 100% /snap/snapd/
/dev/md1          1013688    2564    942416   1% /raid1
/dev/md126        3024752  134144   2717248   5% /raidLinear
/dev/md125        2025360    6144   1898284   1% /raid0
```

➔ fstab 변경 후 재부팅 시 정상 작동 확인

- RAID 6 와 RAID 1+ 0 개념

  - RAID5 보다 신뢰도를 높인 RAID6
  - 신뢰도와 속도 두마리의 토끼를 잡기 위한 RAID1 +0
  - RAID6 은 패리티를 2 개 사용하기 때문에 최소 4 개의 디스크가 필요
  - RAID1+0 은 최소 4 개 디스크가 필요
  - RAID6 의 공간효율은 n-2
  - RAID1+0 의 공간효율은 50%




```
/usr/bin/xauth:  file /home/ubuntu/.Xauthority does not
ubuntu@server:~$ ls -l /dev/sd*
brw-rw---- 1 root disk 8,   0  3월  17 11:39 /dev/sda
brw-rw---- 1 root disk 8,   1  3월  17 11:39 /dev/sda1
brw-rw---- 1 root disk 8,   2  3월  17 11:39 /dev/sda2
brw-rw---- 1 root disk 8,  16  3월  17 11:39 /dev/sdb
brw-rw---- 1 root disk 8,  17  3월  17 11:39 /dev/sdb1
brw-rw---- 1 root disk 8,  32  3월  17 11:39 /dev/sdc
brw-rw---- 1 root disk 8,  33  3월  17 11:39 /dev/sdc1
brw-rw---- 1 root disk 8,  48  3월  17 11:39 /dev/sdd
brw-rw---- 1 root disk 8,  49  3월  17 11:39 /dev/sdd1
brw-rw---- 1 root disk 8,  64  3월  17 11:39 /dev/sde
brw-rw---- 1 root disk 8,  65  3월  17 11:39 /dev/sde1
brw-rw---- 1 root disk 8,  80  3월  17 11:39 /dev/sdf
brw-rw---- 1 root disk 8,  81  3월  17 11:39 /dev/sdf1
brw-rw---- 1 root disk 8,  96  3월  17 11:39 /dev/sdg
brw-rw---- 1 root disk 8,  97  3월  17 11:39 /dev/sdg1
brw-rw---- 1 root disk 8, 112  3월  17 11:39 /dev/sdh
brw-rw---- 1 root disk 8, 113  3월  17 11:39 /dev/sdh1
brw-rw---- 1 root disk 8, 128  3월  17 11:39 /dev/sdi
brw-rw---- 1 root disk 8, 129  3월  17 11:39 /dev/sdi1
brw-rw---- 1 root disk 8, 144  3월  17 11:39 /dev/sdj
brw-rw---- 1 root disk 8, 145  3월  17 11:39 /dev/sdj1
```

➔ 디스크 9 장착 시점으로 스냅샵 이후 RAID 적용 준비 완료

```
/dev/md6          2025360    6144   1898284   1% /raid6
/dev/md10         2023376    6120   1896424   1% /raid10
```

➔ 앞선 내용과 동일하게 RAID 생성 후 마운트, fstab 설정, mdadm.conf 설정 완료

- md6 은 RAID6 으로 기존 RAID 생성과 동일
- md10 은 RAID 0+ 1 으로 RAID1 을 2 개 생성 후 RAID1 들을 묶어 RAID0 생성
- mount 이후 재접속 시 자동 mount 를 위해 fstab 에 등록, mdadm.conf 등록(nameserver 는 삭제)

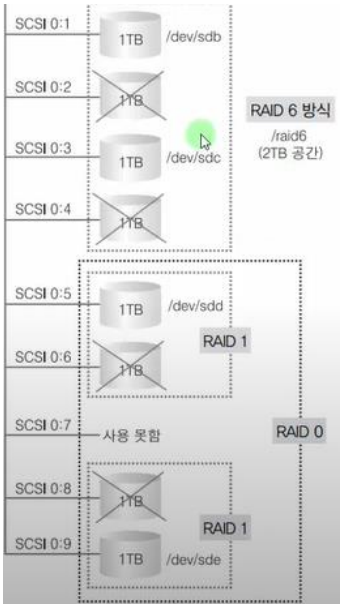
```
root@server:~/바탕화면#
root@server:~/바탕화면# update-initramfs  -u
```

➔ update-initramfs -u 명령어로 업데이트 후 재부팅

```
/dev/md6         2025360    6144   1898284   1% /raid6
/dev/md10        2023376    6120   1896424   1% /raid10
tmpfs             398260      16    398244   1% /run/use
tmpfs             398260       8    398252   1% /run/use
```

➔ RAID 가 잘 적용되어 있는 모습

- RAID6, 10 문제 발생 테스트
  - ➢ 각 2 개씩 하드디스크를 고장 낸다.
  - ➢ 고장 후에도 데이터의 이상 여부를 확인한다.



Processors	1
Hard Disk (SCSI)	80 GB
Hard Disk 9 (SCSI)	1 GB
Hard Disk 2 (SCSI)	2 GB
Hard Disk 6 (SCSI)	1 GB
Hard Disk 4 (SCSI)	1 GB
Hard Disk 10 (SCSI)	1 GB
CD/DVD (SATA)	Auto detect
Network Adapter	NAT
Display	Auto detect

➔ 4 개 하드디스크 삭제

➢ RIAD6 (SCSI0:2, SCSI0:4) RAID10 (SCSI0:6, SCSI0:8) 삭제

```
ubuntu@server:~$ ls /dev/sd*
/dev/sda   /dev/sda2  /dev/sdb1  /dev/sdc1  /dev/sdd1  /dev/sde1  /dev/sdf1
/dev/sda1  /dev/sdb   /dev/sdc   /dev/sdd   /dev/sde   /dev/sdf
ubuntu@server:~$ df
Filesystem     1K-blocks    Used Available Use% Mounted on
udev             1958276       0   1958276   0% /dev
tmpfs             398264    1824    396440   1% /run
/dev/sda2       78106292 7213892  66881752  10% /
tmpfs            1991312       0   1991312   0% /dev/shm
tmpfs               5120       0      5120   0% /run/lock
tmpfs            1991312       0   1991312   0% /sys/fs/cgroup
/dev/loop0           128     128         0 100% /snap/bare/5
/dev/loop1         56320   56320         0 100% /snap/core18/1705
/dev/loop3         56960   56960         0 100% /snap/core18/2284
/dev/loop2        254848  254848         0 100% /snap/gnome-3-38-2004/99
/dev/loop4         63488   63488         0 100% /snap/core20/1376
/dev/loop5         63616   63616         0 100% /snap/gtk-common-themes/150
/dev/loop6        224256  224256         0 100% /snap/gnome-3-34-1804/77
/dev/loop7         66816   66816         0 100% /snap/gtk-common-themes/151
/dev/loop8        246656  246656         0 100% /snap/gnome-3-34-1804/24
/dev/loop9         51072   51072         0 100% /snap/snap-store/433
/dev/loop10        44800   44800         0 100% /snap/snapd/15177
/dev/loop11        55552   55552         0 100% /snap/snap-store/558
tmpfs             398260      16    398244   1% /run/user/0
tmpfs             398260       8    398252   1% /run/user/1000
```

➔ 삭제 후 실행하니 마운트 진행 X (데이터가 사라진 것은 아님)

```
ubuntu@server:~$ sudo mdadm --run /dev/md6
[sudo] ubuntu의 암호 :
mdadm: started array /dev/md6
ubuntu@server:~$ sudo mdadm --run /dev/md2
mdadm: started array /dev/md2
ubuntu@server:~$ sudo mdadm --run /dev/md3
mdadm: started array /dev/md3
ubuntu@server:~$
ubuntu@server:~$ sudo mount /dev/md6 /raid6
ubuntu@server:~$ sudo mount /dev/md10  /raid10
```

➔ 수동 마운트 진행

```
tmpfs             398260      16    398244   1% /run/user/0
tmpfs             398260       8    398252   1% /run/user/1000
/dev/md6         2025360    6144   1898284   1% /raid6
/dev/md10        2023376    6120   1896424   1% /raid10
```

```
ubuntu@server:~$ sudo ls -l /raid6
합계 16
drwx------ 2 root root 16384  3월  17 12:04 lost+found
ubuntu@server:~$ sudo ls -l /raid10
합계 16
drwx------ 2 root root 16384  3월  17 12:09 lost+found
ubuntu@server:~$
```

➔ 디스크가 고장(삭제)된 상태인데도 불구하고 데이터는 보존되어있음

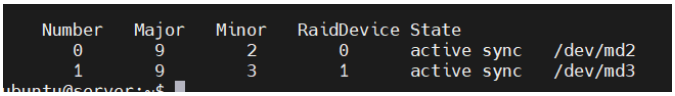
➢ 실제 상황에서는 꼭 데이터를 백업받아야 함 (컴퓨터가 비정상 상태이기 때문)

```
    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       -       0        0        1      removed
       2       8       33        2      active sync   /dev/sdc1
       -       0        0        3      removed
```

➔ /dev/md6 는 /sdb1 , /sdc1 으로만으로 잘 작동되는 것을 확인

```
    Number   Major   Minor   RaidDevice State
       0       8       49        0      active sync   /dev/sdd1
       -       0        0        1      removed
```

```
    Number   Major   Minor   RaidDevice State
       -       0        0        0      removed
       1       8       65        1      active sync   /dev/sde1
```



```
    Number   Major   Minor   RaidDevice State
       0       9       2         0      active sync   /dev/md2
       1       9       3         1      active sync   /dev/md3
```

➔ 각각 md2, md3, md10 상태 확인

➢ md2, 3 는 각각 1 개씩만 작동, md10 은 이상 없음 확인 가능

```
ubuntu@server:~$ sudo umount /dev/md6 /dev/md10
ubuntu@server:~$
ubuntu@server:~$ sudo mdadm --stop /dev/md6
mdadm: stopped /dev/md6
ubuntu@server:~$ sudo mdadm --stop /dev/md10
mdadm: stopped /dev/md10
ubuntu@server:~$ sudo mdadm --stop /dev/md2
mdadm: stopped /dev/md2
ubuntu@server:~$ sudo mdadm --stop /dev/md3
mdadm: stopped /dev/md3
ubuntu@server:~$
```

➔ 정상적인 부팅이 될 수 있도록 RAID 장치 해제

➢ mdadm --stop 을 차례대로 진행하여야 함 (바깥부터/ md10 -> md2,md3)

➢ 이후 이전 학습처럼 원상 복구 절차 진행

● LVM 개념 (1)

➢ LVM (Logical Volume Manage) 개념 이해

✓ LVM 주요 기능

여러 개의 하드디스크를 합쳐서 한 개의 파일시스템으로 사용하는 것으로 필요에 따라서 다시 나눌 수 있다.

예로 2TB 용량의 하드디스크 2 개를 합친 후에 다시 1TB 와 3TB 로 나눠서 사용할 수 있다.

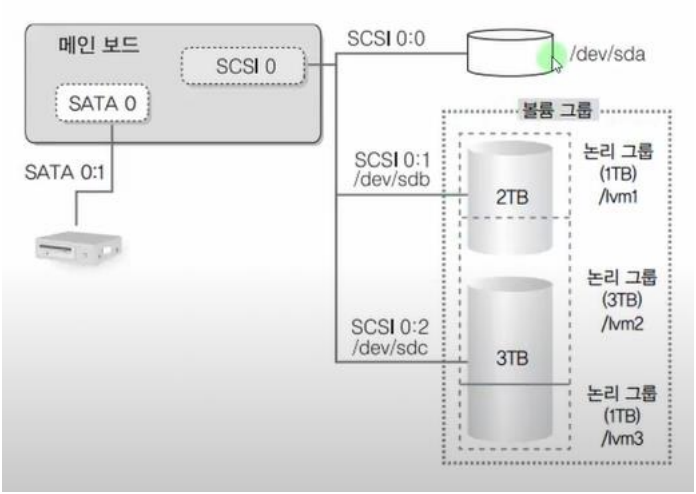
✓ 용어

Physical Volume(물리 볼륨): /dev/sda1, /dev/sab1 등의 파티션

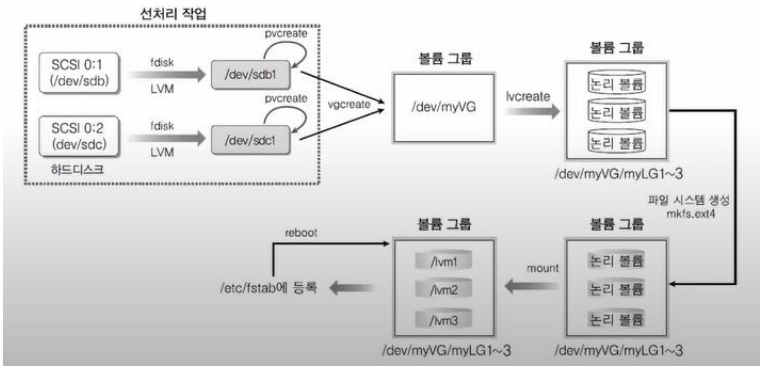
Volume Group(볼륨 그룹): 물리 볼륨을 합쳐서 1 개의 물리 그룹으로 만드는 것

Logical Volume(논리 볼륨): 볼륨 그룹을 1 개 이상으로 나눠서 논리 그룹으로 나눈 것

➢ LVM 을 구현하려고 하드디스크 2 개를 추가한 구성도



● LVM 구현 실습 흐름도



Hard Disk (SCSI)	80 GB
Hard Disk 3 (SCSI)	3 GB
Hard Disk 2 (SCSI)	2 GB
CD/DVD (SATA)	Auto detect
Network Adapter	NAT
Display	Auto detect

➔ 하드디스크 2 개 추가 (3GB, 2GB 총 5GB)

```
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'
```

➔ sdb, sdc fdisk 명령어 작업

➢ type code '8e' 입력하여 LVM 으로 생성

```
ubuntu@server:~$ sudo apt -y install lvm2
패키지 목록을 읽는 중입니다 ... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다 ... 완료
다음의 추가 패키지가 설치될 것입니다 :
  dmeventd libaio1 libdevmapper-event1.02.1 lib
  thin-provisioning-tools
다음 새 패키지를 설치할 것입니다 :
```

➔ 물리 볼륨 명령어인 pvcreate 사용 위해 lvm2 install 진행

```
ubuntu@server:~$ sudo pvcreate /dev/sdb1
  Physical volume "/dev/sdb1" successfully created.
ubuntu@server:~$ sudo pvcreate /dev/sdc1
  Physical volume "/dev/sdc1" successfully created.
```

➔ sdb1 , sdc1 을 물리 볼륨으로 변경

```
ubuntu@server:~$ sudo vgcreate myVG /dev/sdb1 /dev/sdc1
  Volume group "myVG" successfully created
ubuntu@server:~$ vgdisplay
  WARNING: Running as a non-root user. Functionality may be unavailable.
  /run/lock/lvm/P_global:aux: open failed: 허가 거부
ubuntu@server:~$ sudo 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               BEWR0H-LocG-h4wP-T2EV-svkv-RdgJ-l2ROfC
```

➔ 두개의 물리 볼륨을 묶어서 하나의 볼륨 그룹으로 생성

- vgcreate myVG /dev/sdb1 /dev/sdc1 명령어 사용하여 볼륨 그룹 생성
- vgdisplay 명령으로 현재 볼륨 그룹들을 확인 가능

```
ubuntu@server:~$ sudo lvcreate --size 1G --name myLG1 myVG
  Logical volume "myLG1" created.
ubuntu@server:~$ sudo lvcreate --size 3G --name myLG2 myVG
  Logical volume "myLG2" created.
ubuntu@server:~$ sudo lvcreate --extents 100%FREE  --name myLG3  myVG
  Logical volume "myLG3" created.
ubuntu@server:~$
```

➔ 불륨 그룹을 나눠서 논리 그룹 생성

- lvcreate 명령어로 기존 볼륨 그룹을 나눠서 생성

  lvcerate --size 1G --name myLG1 myVG
  | 크기는 ' ' name 은 ' ' [ ] 에서 나눈다

➢ 마지막 그룹을 생성할 때 나머지 공간을 전부 활용

lvcreate --extends 100%FREE --name myLG3 myVG

```
ubuntu@server:~$ sudo ls -l /dev/myVG
합계 0
lrwxrwxrwx 1 root root 7  3월  17 15:00 myLG1 -> ../dm-0
lrwxrwxrwx 1 root root 7  3월  17 15:00 myLG2 -> ../dm-1
lrwxrwxrwx 1 root root 7  3월  17 15:01 myLG3 -> ../dm-2
```

➔ 생산이 성공적으로 완성

```
ubuntu@server:~$ sudo mount /dev/myVG/myLG1  /lvm1
ubuntu@server:~$ sudo mount /dev/myVG/myLG2  /lvm2
ubuntu@server:~$ sudo mount /dev/myVG/myLG3  /lvm3
```

➔ 디렉터리 생성 후 마운트 진행

➢ mkfs.ext4 명령어로 포맷진행

➢ mkdir 명령어로 디렉터리 3 개 생성 (lvm1, 2, 3)

➢ mount 명령어로 마운트 진행

```
13 /dev/myVG/myLG1      /lvm1    ext4   defaults   0   0
14 /dev/myVG/myLG2      /lvm2    ext4   defaults   0   0
15 /dev/myVG/myLG3      /lvm3    ext4   defaults   0   0
```

➔ fstab 수정하여 자동 마운트 설정

```
/dev/mapper/myVG-myLG3   1007640    2544    936696   1% /lvm3
/dev/mapper/myVG-myLG1    999320    2564    927944   1% /lvm1
/dev/mapper/myVG-myLG2   3030800    9216   2847916   1% /lvm2
```

➔ 재부팅 후 정상 작동 확인

● 사용자별 공간 할당 -쿼터

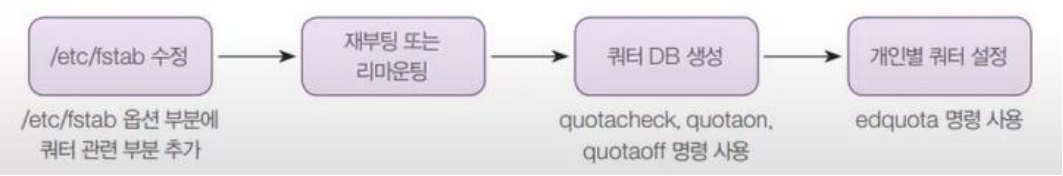
➢ 쿼터(Quota) 개념

✓ 파일시스템마다 사용자나 그룹이 생성할 수 있는 파일의 용량 및 개수를 제한하는 것

✓ 파일시스템을 "/"로 지정하는 것보다는, 별도의 파일시스템을 지정해서 해당 부분을 쓰도록 하는 것이 좋음

✓ "/"파일시스템을 많은 사용자가 동시에 사용하게 되면, 우분투 서버를 운영하기 위해서 디스크를 읽고 쓰는 작업과 일반 사용자가 디스크를 읽고 쓰는 작업이 동시에 발생하므로 전반적으로 시스템의 성능이 저하됨

➢ 사용자를 만들고 해당 사용자에게 공간 할당




```
ubuntu@server:~$ sudo mkfs.ext4 /dev/sdb1
mke2fs 1.45.5 (07-Jan-2020)
Creating filesystem with 2621184 4k blocks and 655360 inodes
Filesystem UUID: 3eaaf812-5e52-4ead-8de0-3c4a0c4c92ae
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

ubuntu@server:~$ sudo mkdir /userhome
ubuntu@server:~$
ubuntu@server:~$ sudo mount /dev/sdb1 /userhome
```

➔ 기존과 동일하게 진행하여 기본 리눅스 타입으로 생성

➢ 포맷진행 > 디렉터리생성 > 마운트 > fstab 등록

```
ubuntu@server:~$ sudo adduser --home  /userhome/john  john
```

```
ubuntu@server:~$ sudo adduser --home  /userhome/daniel daniel
```

➔ adduser 명령어로 사용자 2 개 추가 (john , daniel // passwd 는 동일하게 '1234')

➢ adduser --home /userhome/john john

```
12
13 /dev/userhome    ext4    defaults,usrjquota=aquota=aquita.user, jpfmt=vfsv0  0    0
```

➔ 생성한 일반 디스크를 쿼터용으로 변경

- defaults,usrjqota=auqota=aquta.user, jpfmt=vfsv0 추가

```
ubuntu@server:~$ sudo mount --options remount /userhome/
ubuntu@server:~$
```

➔ remount 로 재부팅 효과

- 재부팅을 해도 좋지만 remount 로 재부팅의 효과를 줄 수 있음 (다시 마운트함)
- mount --options remount [디렉터리명]

```
/dev/sdb1 on /userHome type ext4 (rw,relatime,jqfmt=vfsv0,usrjquota=aquota.user)
root@server:~/바탕화면#
```

➔ 쿼터형으로 마운트가 성공적으로 적용된 것을 확인

```
ubuntu@server:~$ sudo apt install quota
패키지 목록을 읽는 중입니다 ... 완료
```

➔ 쿼터 패키지 install 진행

```
/dev/sdb1 [/userHome]: user quotas turned off
root@server:/userHome# quotacheck -augmn
root@server:/userHome# rm -f aquota.*
root@server:/userHome# quotacheck -augmn
root@server:/userHome# touch aquota.user aquota.group
root@server:/userHome# chmod 600 aquota.*
root@server:/userHome#
root@server:/userHome# quotacheck -augmn

root@server:/userHome# quotaon -avug
/dev/sdb1 [/userHome]: user quotas turned on
```

➔ 쿼터 작동 성공

```
 GNU nano 4.8                          /tmp//EdP.aq76Bf7
Disk quotas for user john (uid 1001):
  Filesystem                blocks       soft       hard     inodes     soft     hard
  /dev/sdb1                     16      30720      40960          4        0        0
```

➔ 사용자 공간 제어

- edquota -u john 명령어로 john 사용자 설정
- blocks, soft, hard 로 공간을 제어할 수 있다. (단위는 kb) // 0 은 무제한으로 사용 가능
- soft 는 초과 할 수는 있지만 경고 / hard 는 절대로 초과할 수 없다

```
root@server:~# su - john
john@server:~$ whoami
john
john@server:~$ pwd
/userHome/john
```

➔ 사용자 공간 제한 테스트를 위해 john 으로 접속

```
john@server:~$ cp   /boot/vmlinuz-*  test1
john@server:~$ cp   /boot/vmlinuz-*  test2
john@server:~$ cp   /boot/vmlinuz-*  test3
john@server:~$ ls  -l
합계 34164
-rw-r--r-- 1 john john 11657976  1월 10 12:16 test1
-rw-r--r-- 1 john john 11657976  1월 10 12:16 test2
-rw-r--r-- 1 john john 11657976  1월 10 12:16 test3
john@server:~$ cp   /boot/vmlinuz-*  test4
cp: 'test4'에 쓰는 도중 오류 발생: 디스크 할당량이 초과됨
john@server:~$
```

➔ cp 명령어가 진행되다가 공간 제한에 걸리자 실행 취소가 된다

- quota 명령어로 사용자는 자신의 할당량을 확인 가능
  (quota = soft / limit = hard)

```
root@server:/userHome# repquota /userHome/
*** Report for user quotas on device /dev/sdb1
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      --      20       0       0              3     0     0
john      --   11420   30720   40960              9     0     0
daniel    --      16       0       0              4     0     0
```

➔ root 사용자는 repqouta [디렉토리명]명령어로 사용자 별 사용량 확인 가능

- 셸의 기본과 작성법
- 우분투의 bash 셸 (터미널)
  - 기본 셸은 bash(Bourne Again Shell:'배시 셸')
  - bash 셸의 특징
    - ✓ Alias 기능(명령어 단축 기능)
    - ✓ History 기능(위/아래 화살표키)
    - ✓ 연산 기능
    - ✓ job Control 기능
    - ✓ 자동 이름 완성 기능 (탭키)
    - ✓ 프롬프트 제어 기능
    - ✓ 명령 편집 기능
  - 셸의 명령문 처리 방법
    - ✓ (프롬프트) 명령어 [옵션...][인자...]
    - ✓ 예) # rm -fg /mydir

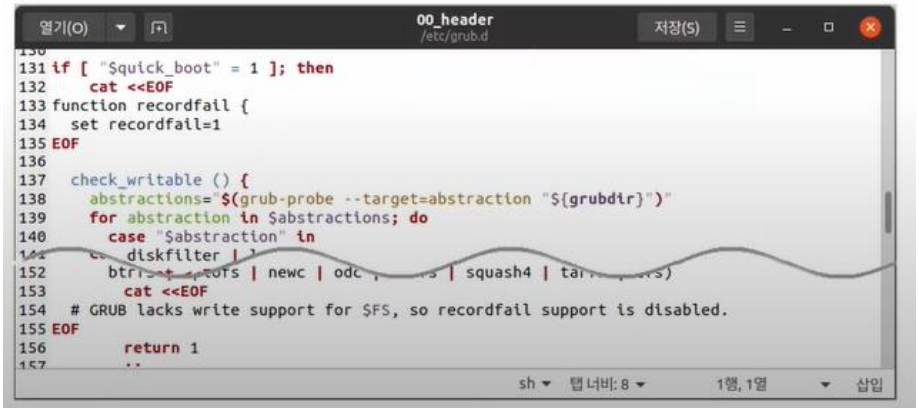
- 환경 변수
  - "echo $환경변수이름" 으로 확인 가능
  - "export 환경변수=값" 으로 환경 변수의 값을 변경
  - 주요 환경변수

환경 변수	설명	환경 변수	설명
HOME	현재 사용자의 홈 디렉터리	PATH	실행 파일을 찾는 디렉터리 경로
LANG	기본 지원되는 언어	PWD	사용자의 현재 작업 디렉터리
TERM	로그인 터미널 타입	SHELL	로그인해서 사용하는 셸
USER	현재 사용자의 이름	DISPLAY	X 디스플레이 이름
COLUMNS	현재 터미널의 컬럼 수	LINES	현재 터미널 라인 수
PS1	1차 명령 프롬프트 변수	PS2	2차 명령 프롬프트(대개는 '>')
BASH	bash 셸의 경로	BASH_VERSION	bash 버전
HISTFILE	히스토리 파일의 경로	HISTSIZE	히스토리 파일에 저장되는 개수
HOSTNAME	호스트의 이름	USERNAME	현재 사용자 이름
LOGNAME	로그인 이름	LS_COLORS	ls 명령의 확장자 색상 옵션
MAIL	메일을 보관하는 경로	OSTYPE	운영체제 타입

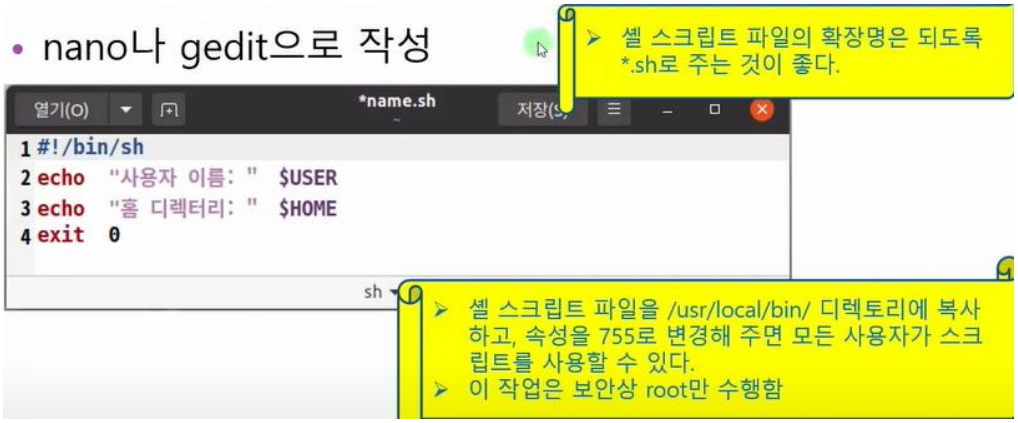
```
root@server:~/바탕화면# echo $HOME
/root
root@server:~/바탕화면# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/lo
cal/games:/snap/bin
root@server:~/바탕화면#
```

➔ HOME 과 PATH 사용 확인

- 셸 스크립트 프로그래밍
  - c 언어와 유사하게 프로그래밍이 가능
  - 변수, 반복문, 제어문 등의 사용이 가능
  - 별도로 컴파일하지 않고 텍스트 파일 형태로 바로 실행
  - vi 나 gedit 으로 작성이 가능
  - 리눅스의 많은 부분이 셸 스크립트로 작성되어 있음



- 셸 스크립트의 작성과 실행



➢ 실행방법

1. "sh <스크립트파일>"로 실행

2. "chmod +x <스크립트 파일>" 명령으로 실행 가능 속성으로 변경한 후에, "./<스크립트파일>" 명령으로 실행

```
1 #!/bin/sh
2 echo  "사용자 이름: " $USER
3 echo "홈 디렉터리: " $HOME
4 exit 0
```

```
root@server:~# ls -l name.sh
-rw-r--r-- 1 root root 82  3월 17 17:58 name.sh
root@server:~# sh name.sh
사용자 이름:  root
홈 디렉터리:  /root
root@server:~#
```

➔ gedit 로 name.sh 을 생성 뒤 작성 후 실행

➢ 사용자이름과 홈 디렉터리가 나타나는 기능 코딩

➢ sh <스크립트파일> 로 실행 (sh name.sh)