Chapter 07 파일 시스템과 디스크 관리

목차

학습목표

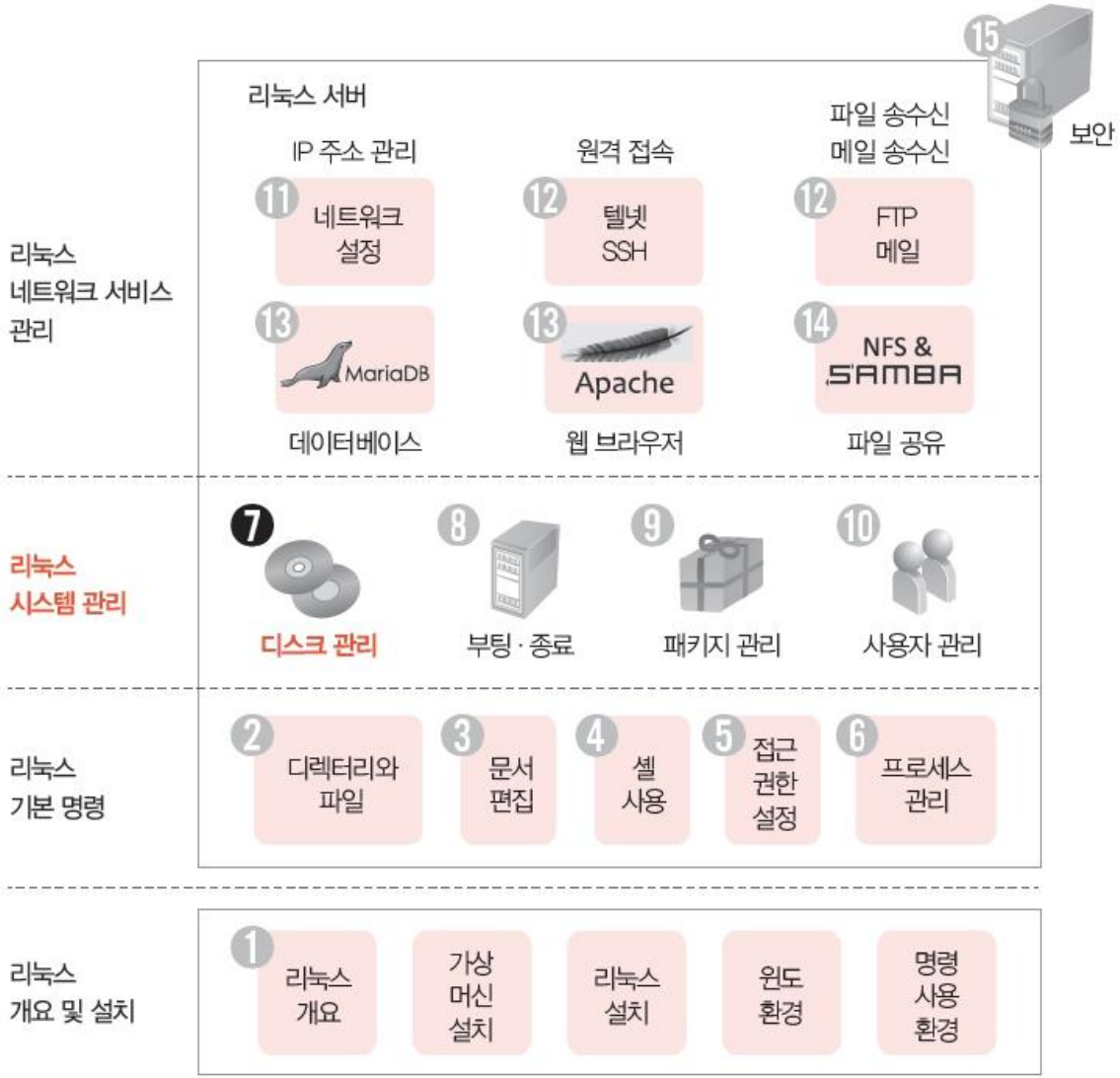
- 리눅스에서 지원하는 파일 시스템의 종류와 구조를 설명할 수 있다.
- 마운트의 개념을 이해하고 설명할 수 있다.
- 이동식 장치를 마운트하여 사용할 수 있다.
- LVM의 개념을 이해하고 설명할 수 있다.
- 파일 시스템을 구축하고 관리하는 방법을 설명할 수 있다.

00 개요

00. 개요

■ 리눅스 학습 맵에서 7장의 위치

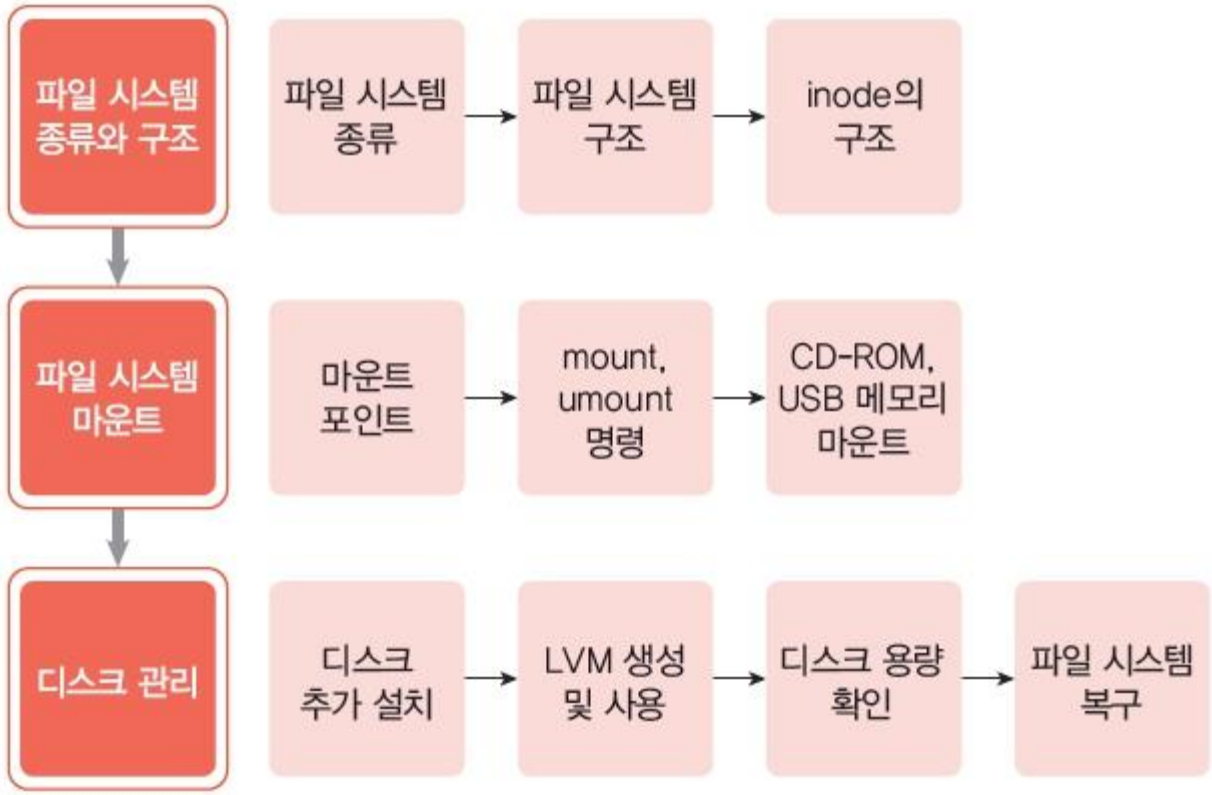
- 7장은 제3단계 중 첫 항목으로 파일 시스템과 디스크를 다룬다.
- 시스템에 설치된 디스크의 파일 시스템을 관리하는 방법과 새 디스크를 추가하고 파일 시스템을 생성하여 사용자가 사용할 수 있도록 하는 방법을 익힌다.



00. 개요

7장의 내용 구성

- 파일 시스템의 종류와 구조 이해하기
- inode의 구조 이해하기
- 파일 시스템을 마운트하고 언마운트 하는 방법을 익히기
- 디스크를 추가로 설치하고 파일 시스템을 생성하는 방법을 익히기
- 파일 시스템을 LVM으로 변환하는 방법 익히기
- 디스크 용량을 확인하고 파일 시스템을 복구하는 방법 익히기



01 리눅스 파일 시스템 종류

01. 리눅스 파일 시스템 종류

디스크 기반 파일 시스템

- ext(ext1): 1992년 발표, 시스템 최대 크기 2GB, 현재는 사용하지 않음
- ext2: 1993년 발표, ext1의 문제를 해결하고 확장성 강화, 시스템 최대 크기 2TB(이론적으로는 32TB)
- ext3: 2001년 발표, ext2와 호환, 저널링 기능 도입(디스크에 기록되는 데이터의 복구 기능 강화), 시스템 최대 크기 2TB~32TB
- ext4: 2008년 발표, 1EB 이상의 볼륨과 16TB 이상의 파일 지원, ext2/ext3와 호환, 서브디렉터리 최대 64000개까지 가능
- XFS: 1993년 실리콘 그래픽스가 개발, 2001년 리눅스에 이식, 64비트 파일시스템으로 최대 16EB까지 지원

01. 프로세스의 개념

■ 리눅스에서 지원하는 다양한 파일 시스템

표 7-1 리눅스에서 지원하는 기타 파일 시스템

파일 시스템	기능
msdos	MS-DOS 파티션을 사용하기 위한 파일 시스템이다.
iso9660	CD-ROM, DVD의 표준 파일 시스템이며 읽기 전용으로 사용된다.
nfs	Network File System으로 원격 서버의 디스크를 연결할 때 사용된다.
ufs	Unix File System으로 유닉스의 표준 파일 시스템이다.
vfat	윈도 95, 98, NT를 지원하기 위한 파일 시스템이다.
hpfs	HPFS를 지원하기 위한 파일 시스템이다.
ntfs	윈도의 NTFS를 지원하기 위한 파일 시스템이다.
sysv	유닉스 시스템V를 지원하기 위한 파일 시스템이다.
hfs	맥 컴퓨터의 hfs 파일 시스템을 지원하기 위한 파일 시스템이다.

01. 프로세스의 개념

■ 특수 용도의 가상 파일 시스템

표 7-2 리눅스의 가상 파일 시스템

파일 시스템	기능
swap	• 스왑 영역을 관리하기 위한 스왑 파일 시스템이다.
tmpfs	• Temporary File System으로 메모리에 임시 파일을 저장하기 위한 파일 시스템이며, 시스템이 재시작할 때마다 기존 내용이 없어진다. • /tmp 디렉터리를 예로 들 수 있다.
proc	• proc 파일 시스템으로 /proc 디렉터리다. • 커널의 현재 상태를 나타내는 파일을 가지고 있다.
ramfs	• 램디스크를 지원하는 파일 시스템이다.
rootfs	• Root File System으로 / 디렉터리다. • 시스템 초기화 및 관리에 필요한 내용을 관리한다.

01. 프로세스의 개념

■ 현재 시스템이 지원하는 파일 시스템 확인

- /proc/filesystems 파일에 현재 시스템이 지원하는 파일 시스템의 종류 저장

```
user1@myubuntu:~$ cat /proc/filesystems
nodev   sysfs
nodev   tmpfs
nodev   bdev
nodev   proc
nodev   cgroup
nodev   cgroup2
nodev   cpuset
nodev   devtmpfs
```

02 리눅스 파일 시스템 구조

02. 리눅스 파일 시스템 구조

ext4 파일 시스템의 구조

- 저장 장치를 논리적인 블록의 집합(블록 그룹)으로 구분
- 일반적으로 블록은 4KB 크기이고 조정 가능

- 블록 그룹 개수 = 장치 크기 /블록 그룹의 크기

02. 리눅스 파일 시스템 구조

- 블록 그룹 유형
  - 블록 그룹 0: 파일 시스템의 첫 번째 블록 그룹으로 특별하게 그룹 0 패딩과 슈퍼블록, 그룹 디스크립터로 구성
  - 블록 그룹 a: 파일 시스템에서 첫 번째 블록 그룹이 아닌 블록 그룹으로 그룹 0 패딩이 없으나 슈퍼블록과 그룹 디스크립터의 복사본을 가지고 있음
  - 블록 그룹 b: 파일 시스템에서 첫 번째 블록 그룹이 아닌 블록 그룹으로 바로 데이터 블록 비트맵으로 시작

그림 7-2 ext4 파일 시스템의 구조

02. 리눅스 파일 시스템 구조

- 그룹 0 패딩
  - 블록 그룹 0의 첫 1,024B는 특별한 목적으로 사용되는데, x86 부트 섹터와 부가 정보를 저장
- 슈퍼블록: 파일 시스템의 다양한 정보 저장, 복구용 백업 슈퍼블록 있음
  - 전체 inode의 개수
  - 할당되지 않은 블록free block의 개수
  - 첫 번째 데이터 블록의 주소
  - 그룹당 블록의 개수
  - 파일 시스템의 상태
  - 전체 블록의 개수
  - 할당되지 않은 inodefree inode의 개수
  - 블록의 크기
  - 마운트 시간
  - 그룹 디스크립터의 크기

02. 리눅스 파일 시스템 구조

- 그룹 디스크립터와 GDT 예약 블록
  - 블록 그룹 0에서 슈퍼블록의 다음에 위치하며 다음 정보 저장
- 데이터 블록 비트맵과 inode 비트맵
- inode 테이블
- 데이터 블록: 실제 정보 저장, 1KB~8KB

- 블록 비트맵의 주소
- inode 비트맵의 주소
- inode 테이블의 주소
- 할당되지 않은 블록의 개수
- 할당되지 않은 inode의 개수
- 디렉터리의 개수
- 블록 비트맵, inode 비트맵 체크섬

02. 리눅스 파일 시스템 구조

- inode 구조

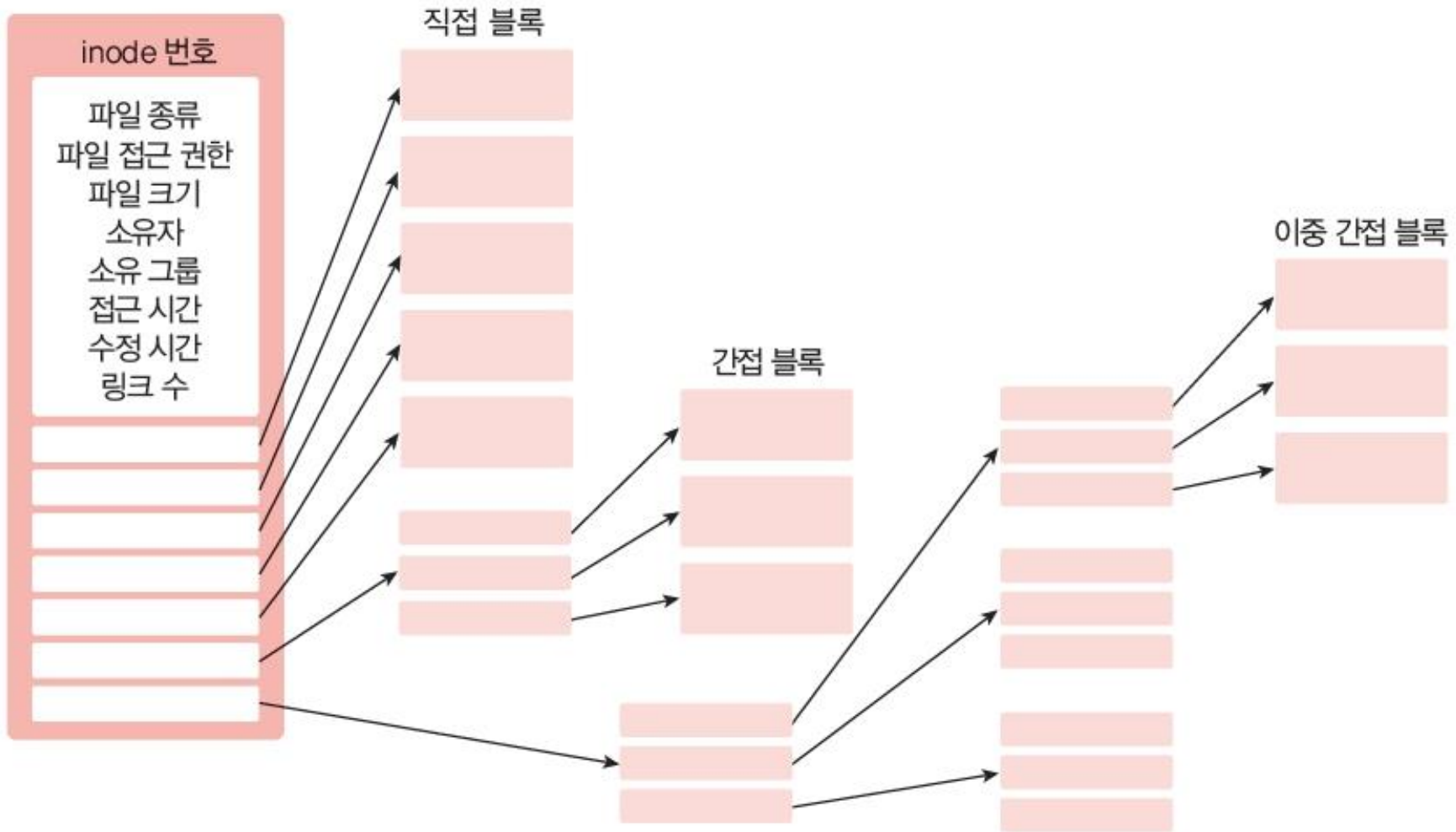


그림 7-3 inode의 구조

02. 리눅스 파일 시스템 구조

■ 파일 시스템과 디렉터리 계층 구조

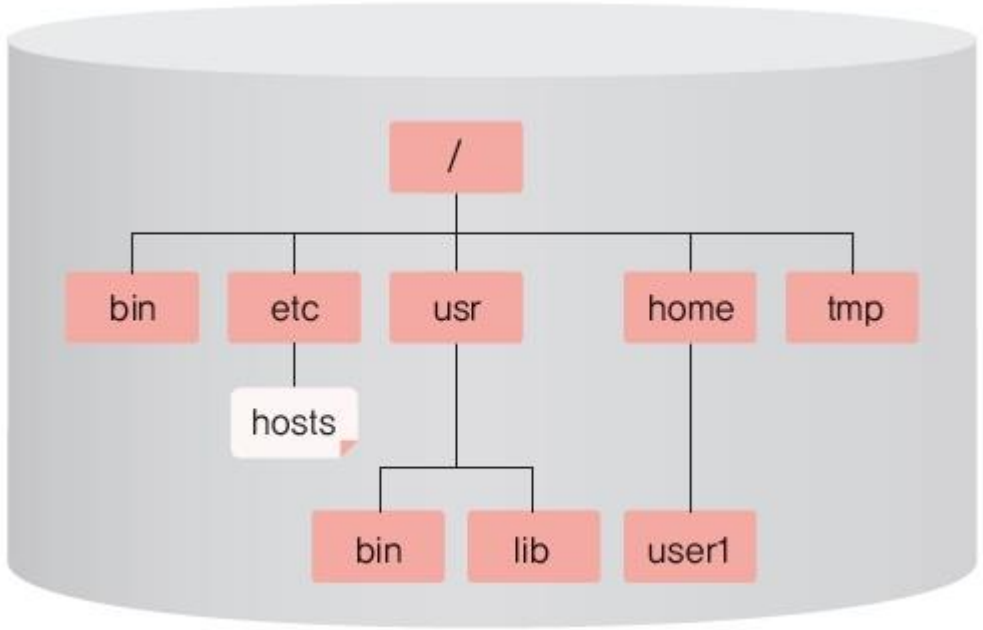


그림 7-4 한 파일 시스템으로 구성하기

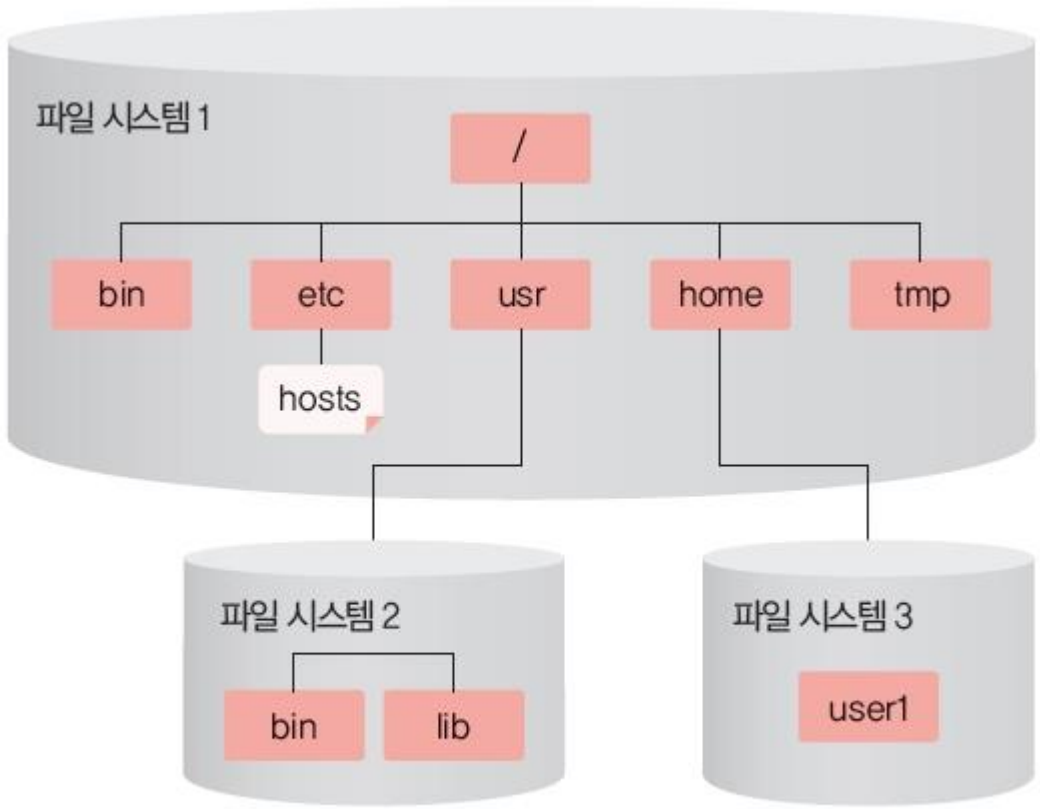


그림 7-5 여러 파일 시스템으로 구성하기

03 파일 시스템 마운트

03. 파일 시스템 마운트

마운트(mount)

- 파일 시스템을 디렉터리 계층 구조의 특정 디렉터리와 연결하는 것
- 파일 시스템이 디렉터리 계층 구조와 연결되지 않으면 사용자가 해당 파일 시스템에 접근할 수 없음

마운트 포인트

- 디렉터리 계층 구조에서 파일 시스템이 연결되는 디렉터리

03. 파일 시스템 마운트

■ 파일 시스템 마운트 설정 파일: /etc/fstab

- 파일 시스템의 마운트 설정 정보 wjwkd
- 부팅시 이 파일을 읽고 파일 시스템 마운트

- /etc/fstab 파일의 구조

그림 7-6 /etc/fstab 파일의 구조

03. 파일 시스템 마운트

/etc/fstab 파일 예

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda3 during installation
UUID=265c8913-ee1e-4034-885f-207969b0a23b /               ext4    errors=remount-
ro 0       1
# /boot/efi was on /dev/sda2 during installation
UUID=A7A5-E2E2  /boot/efi       vfat    umask=0077      0       1
/swapfile                       none            swap    sw              0       0
```

03. 파일 시스템 마운트

```
UUID=265c8913-ee1e-4034-885f-207969b0a23b /       ext4    errors=remount-ro  0       1
```

❶ 장치명: UUID=265c8913-ee1e-4034-885f-207969b0a23b
(UUID: 파일 시스템을 유일하게 구분해주는 128bit 숫자)
❷ 마운트 포인트: /
❸ 파일 시스템 종류: ext4
❹ 옵션: errors=remount-ro
❺ 덤프 관련 설정: 0 (0: dump 명령으로 덤프 안됨 1: 덤프 가능)
❻ 파일 점검 옵션: 1
(0: 부팅시 fsck명령으로 파일시스템 점검 안함,
1: 루트 파일시스템, 2: 루트 외 파일 시스템)

03. 파일 시스템 마운트

표 7-3 파일 시스템의 마운트 옵션

옵션	의미
defaults	일반적인 파일 시스템에 지정하는 속성이다. rw, nouser, auto, exec, suid 속성을 모두 포함한다.
auto	부팅 시 자동으로 마운트한다.
exec	실행 파일이 실행되는 것을 허용한다.
suid	setuid, setgid의 사용을 허용한다.
ro	읽기 전용 파일 시스템이다.
rw	읽기, 쓰기가 가능한 파일 시스템이다.
user	일반 사용자도 마운트가 가능하다.
nouser	일반 사용자의 마운트가 불가능하다. root만 마운트할 수 있다.
noauto	부팅 시 자동으로 마운트하지 않는다.
noexec	실행 파일이 실행되는 것을 허용하지 않는다.
nosuid	setuid, setgid의 사용을 금지한다.
usrquota	사용자별로 디스크 쿼터 설정이 가능하다.
grpquota	그룹별로 디스크 쿼터 설정이 가능하다.

03. 파일 시스템 마운트

■ 마운트 관련 명령

mount

- **기능** 파일 시스템을 마운트한다.
- **형식** mount [옵션] [장치명 마운트 포인트]
- **옵션** -t 파일 시스템 종류: 파일 시스템 종류를 지정한다.
  -o 마운트 옵션: 마운트 옵션을 지정한다.
  -f: 마운트를 할 수 있는지 점검만 한다.
  -r: 읽기만 가능하게 마운트한다(-o ro와 동일).
- **사용 예**

```
mount
mount /dev/sdb1 /
mount -t iso9660 /dev/cdrom /mnt/cdrom
```

03. 파일 시스템 마운트

■ 마운트 해제(언마운트) 명령

umount

- **기능** 파일 시스템을 언마운트한다.
- **형식** umount [옵션] 장치명 또는 마운트 포인트
- **옵션** -t 파일 시스템 종류: 파일 시스템 종류를 지정한다.
- **사용 예** umount /dev/sdb1
  umount /mnt

03. 파일 시스템 마운트

- **mount 명령만 사용하는 경우: 현재 마운트되어 있는 정보 출력**
  - /etc/mtab 파일 내용과 동일

```
user1@myubuntu:~$ mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=1968024k,nr_
inodes=492006,mode=755,inode64)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmo
de=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,noexec,relatime,size=398672k,mode=755,ino
de64)
/dev/sda3 on / type ext4 (rw,relatime,errors=remount-ro)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,inode64)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k,inode64)
(생략)
```

03. 파일 시스템 마운트

/etc/mtab 파일

```
user1@myubuntu:~$ cat /etc/mtab
sysfs /sys sysfs rw,nosuid,nodev,noexec,relatime 0 0
proc /proc proc rw,nosuid,nodev,noexec,relatime 0 0
udev /dev devtmpfs rw,nosuid,relatime,size=1968024k,nr_
inodes=492006,mode=755,inode64 0 0
devpts /dev/pts devpts rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000 0 0
tmpfs /run tmpfs rw,nosuid,nodev,noexec,relatime,size=398672k,mode=755,inode64 0 0
/dev/sda3 / ext4 rw,relatime,errors=remount-ro 0 0
securityfs /sys/kernel/security securityfs rw,nosuid,nodev,noexec,relatime 0 0
tmpfs /dev/shm tmpfs rw,nosuid,nodev,inode64 0 0
tmpfs /run/lock tmpfs rw,nosuid,nodev,noexec,relatime,size=5120k,inode64 0 0
(생략)
```

❶ 장치명
❷ 마운트 포인트
❸ 파일 시스템의 종류
❹ 옵션
❺ 사용하지 않는 항목 두 개(0 0)

03. 파일 시스템 마운트

■ 장치 연결하기: mount

```
mount /dev/sdb1 /mnt
```

장치명

마운트포인트

표 7-4 다양한 장치 마운트의 예

장치	mount 명령 형식의 예
ext2 파일 시스템	mount –t ext2 /dev/sdb1 /mnt
ext3 파일 시스템	mount –t ext3 /dev/sdb1 /mnt
ext4 파일 시스템	mount –t ext4 /dev/sdb1 /mnt mount /dev/sdb1 /mnt
CD-ROM	mount –t iso9660 /dev/cdrom /mnt/cdrom
윈도 디스크	mount –t vfat /dev/hdc /mnt
USB 메모리	mount /dev/sdc1 /mnt → 리눅스용 USB 메모리의 경우 mount –t vfat /dev/sdc1 /mnt → 윈도용 USB 메모리의 경우
읽기 전용 마운트	mount –r /dev/sdb1 /mnt
읽기/쓰기 마운트	mount –w /dev/sdb1 /mnt
원격 디스크 마운트	mount –t nfs 서버 주소:/NFS 서버 측 디렉터리 /mnt

334p. 따라해보기 : 리눅스용 USB 메모리 연결하기

① USB 메모리를 리눅스 시스템에 인식시키기

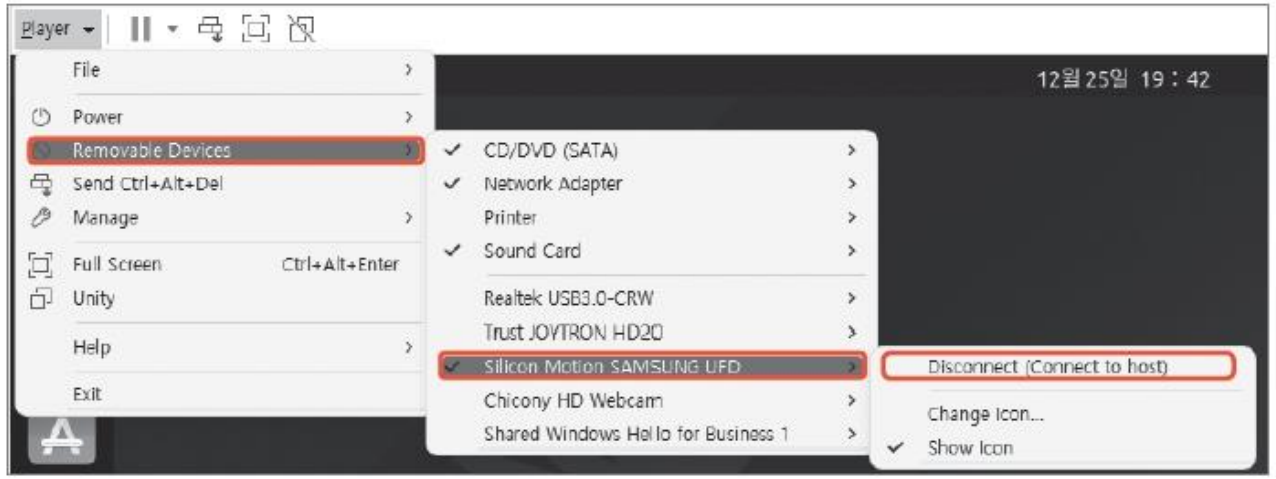


그림 7-7 USB 메모리 스틱 연결하기

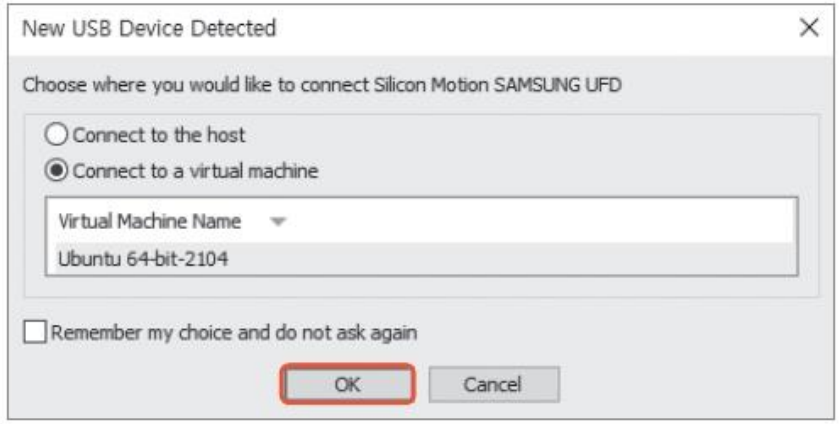


그림 7-8 USB 메모리 연결 메시지

- mount 명령을 실행해보면 장치가 추가되었는지 확인하고 마운트 해제

```
user1@myubuntu:~$ mount
(생략)
/dev/sdb1 on /media/user1/49272dc1-1806-493f-bc1f-d44d654c5bbe type ext4 (rw,nosuid,n
odev,relatime,uhelper=udisks2)
```

```
user1@myubuntu:~$ umount /dev/sdb1
```

334p. 따라해보기 : 리눅스용 USB 메모리 연결하기

② USB 메모리의 장치명 확인하기: sudo fdisk -l

③ USB 메모리에 파티션 생성하기

```
user1@myubuntu:~$ sudo fdisk /dev/sdb


Welcome to fdisk (util-linux 2.36.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

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

334p. 따라해보기 : 리눅스용 USB 메모리 연결하기

```
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-15532031, default 2048):                    → Enter를 입력한다.

Last sector, +sectors or +size{K,M,G,T,P} (2048-15532031, default 15532031):
                                                               → Enter를 입력한다.

Created a new partition 1 of type 'Linux' and of size 7.4 GiB.
Partition #1 contains a ext4 signature.

Do you want to remove the signature? [Y]es/[N]o: Y

The signature will be removed by a write command.

Command (m for help)
```

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

334p. 따라해보기 : 리눅스용 USB 메모리 연결하기

④ 생성한 파티션 포맷하고 파일 시스템 생성하기

```
user1@myubuntu:~$ sudo mke2fs -t ext4 /dev/sdb1
mke2fs 1.46.3 (27-Jul-2021)
Creating filesystem with 1941248 4k blocks and 485760 inodes
```

```
Filesystem UUID:  8eb696ef-924c-4b6b-ac43-e5fc73bbcea9
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      → done이 나올 때까지 기다린다.
Writing superblocks and filesystem accounting information: done
                                           → done이 나올 때까지 기다린다.
```

334p. 따라해보기 : 리눅스용 USB 메모리 연결하기

⑤ USB 파일 시스템 마운트하기

```
user1@myubuntu:~$ sudo mount /dev/sdb1 /mnt
user1@myubuntu:~$ mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
(생략)
/dev/sdb1 on /mnt type ext4 (rw,relatime)
```

⑥ 디렉터리 사용하기

```
user1@myubuntu:~$ cd /mnt
user1@myubuntu:/mnt$
lost+found
```

```
user1@myubuntu:/mnt$ cp /etc/hosts .
cp: 일반 파일 './hosts'을(를) 생성할 수 없음: 허가 거부
user1@myubuntu:/mnt$ sudo cp /etc/hosts .
user1@myubuntu:/mnt$ ls
hosts  lost+found
```

03. 파일 시스템 마운트

■ 장치 연결 해제하기: umount

```
user1@myubuntu:/mnt$ sudo umount /mnt
umount: /mnt: target is busy.
```

busy: 사용중이라는 의미

```
user1@myubuntu:/mnt$ cd
user1@myubuntu:~$ sudo umount /mnt
user1@myubuntu:~$
```

342p. 따라해보기 : 윈도용 USB 메모리 연결하기

① USB 메모리를 USB 슬롯에 꽂고 리눅스 시스템에 인식시키기

② USB 메모리의 장치명을 확인: fdisk -l

```
user1@myubuntu:~$ sudo fdisk -l
(생략)

Disk /dev/sdb: 7.41 GiB, 7952400384 bytes, 15532032 sectors
Disk model: UFD
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: 0x00000000

Device     Boot Start      End  Sectors  Size Id Type
/dev/sdb1  *       80  15532031 15531952  7.4G  c W95 FAT32 (LBA)
```

342p. 따라해보기 : 윈도용 USB 메모리 연결하기

③ USB 메모리를 마운트: FAT32(vfat)

```
user1@myubuntu:~$ sudo mount -t vfat /dev/sdb1 /mnt
user1@myubuntu:~$ mount
(생략)
/dev/sdb1 on /mnt type vfat (rw,relatime,fmask=0022,dmask=0022,codepage=437,iochars
et=iso8859-1,shortname=mixed,errors=remount-ro)
```

④ USB 메모리가 디렉터리에 연결되었으므로 사용 가능

```
user1@myubuntu:~$ cd /mnt
user1@myubuntu:/mnt$ ls
'System Volume Information'  파일
```

```
user1@myubuntu:/mnt$ sudo cp /etc/hosts .
user1@myubuntu:/mnt$ ls
'System Volume Information'  hosts  파일
```

⑤ USB 메모리를 사용하고 나면 umount 명령으로 마운트를 해제

344p. 따라해보기 : CD-ROM/DVD 연결하기

① **CD-ROM/DVD 장치를 연결하여 리눅스 시스템에 인식시킴**

- 대부분의 경우 CD-ROM/DVD는 자동으로 인식

② **수동으로 CD-ROM/DVD를 마운트할 경우 장치명은 /dev/cdrom이나 /dev/sr#을 사용**

```
user1@myubuntu:~$ sudo umount /dev/sr1
user1@myubuntu:~$ sudo mount -r -t iso9660 /dev/sr1 /mnt
```

③ **CD-ROM/DVD가 마운트되었으므로 내용을 확인하고 사용**

④ **CD-ROM 장치를 사용하고 나면 umount 명령을 사용하여 마운트를 해제**

04 디스크 추가 설치

04. 디스크 추가 설치

■ 디스크 추가 단계

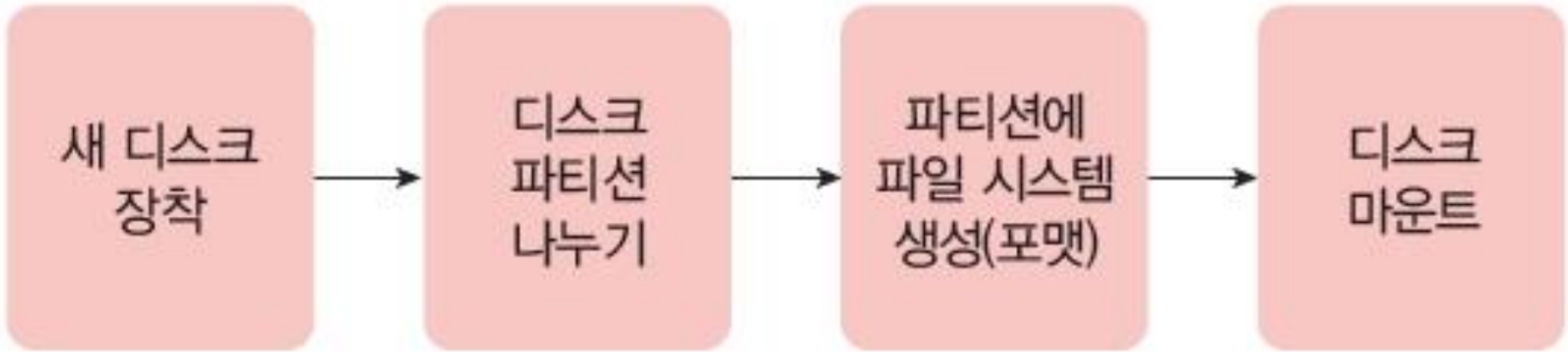


그림 7-10 디스크 추가 단계

04. 디스크 추가 설치

■ 가상머신에 디스크를 추가하는 방법

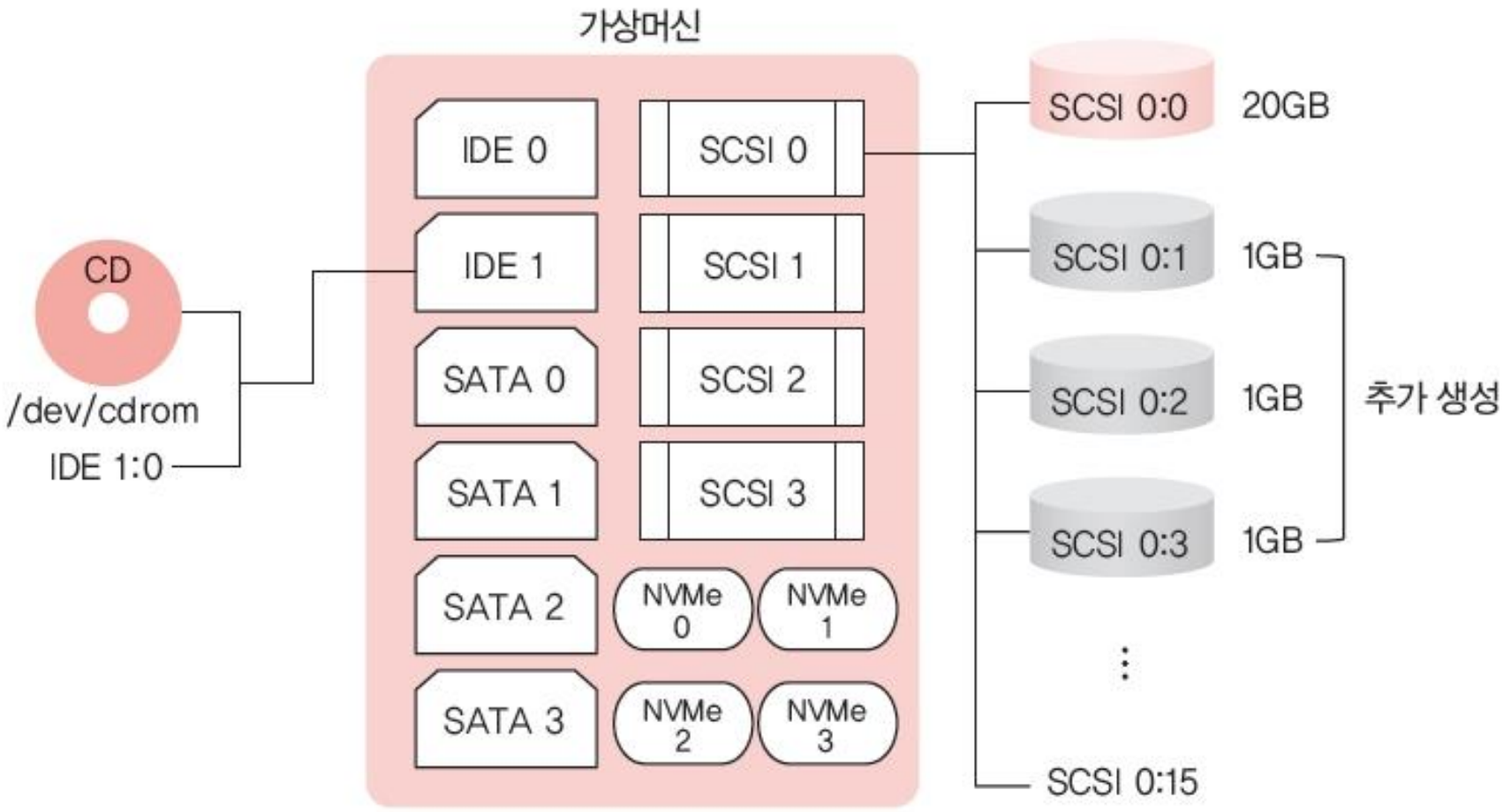
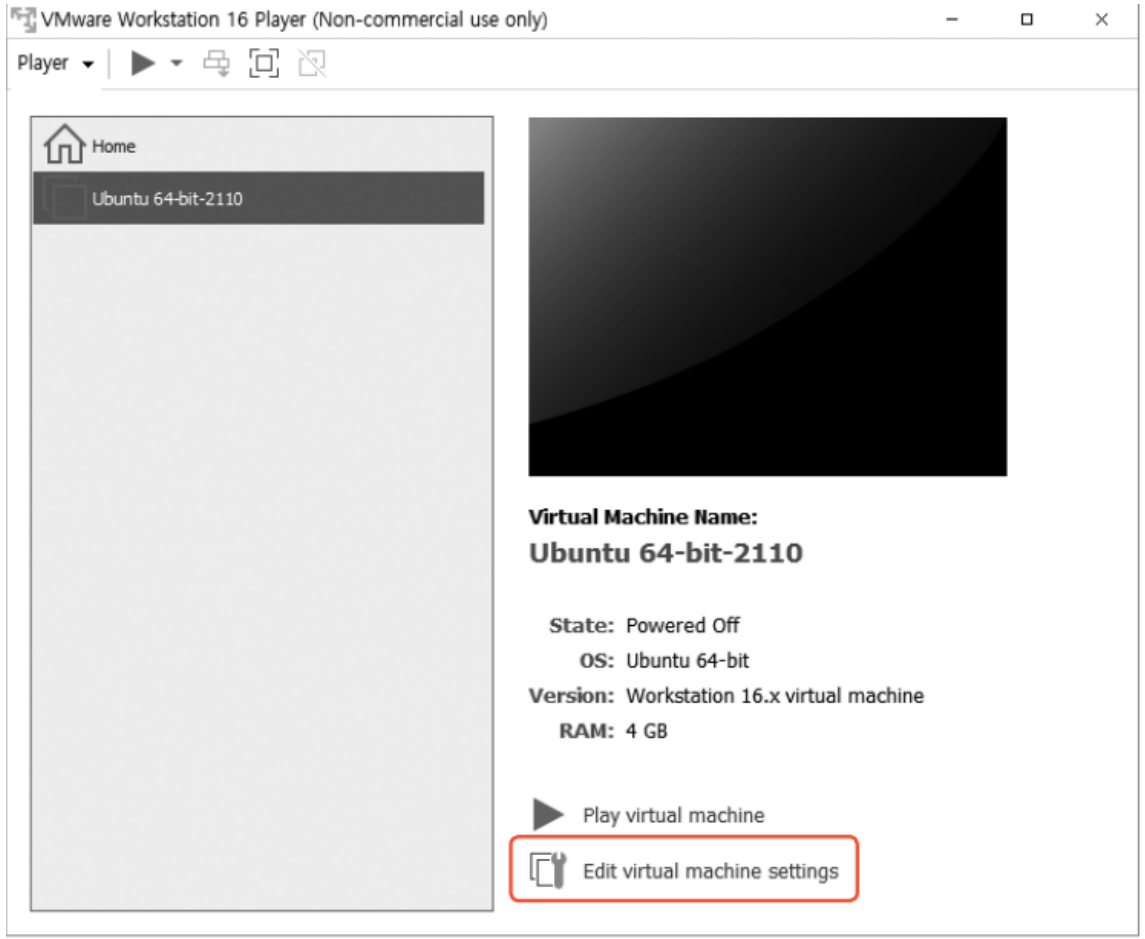


그림 7-12 가상머신의 디스크 추가 구성 개념도

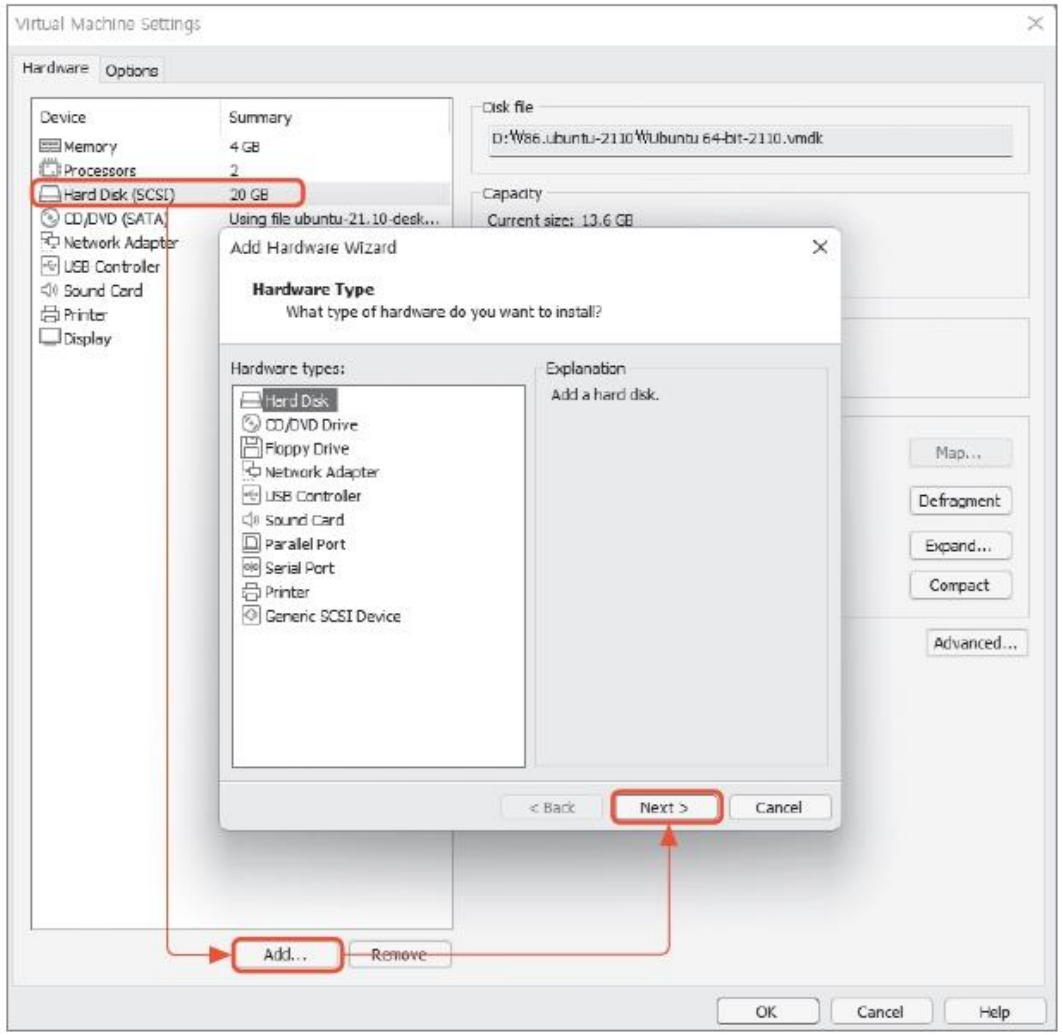
348p. 따라해보기 : 가상머신에 디스크 추가 장착하기

① Player 메뉴에서 [Manage]-[Virtual Machine Settings]를 선택 또는
VMware Player 메인 화면에서 [Edit virtual machine settings]를 선택



348p. 따라해보기 : 가상머신에 디스크 추가 장착하기

② Virtual Machine Settings 창에서 [Add...]를 선택-> Add Hardware Wizard 창 -> [Next]를 클릭



348p. 따라해보기 : 가상머신에 디스크 추가 장착하기

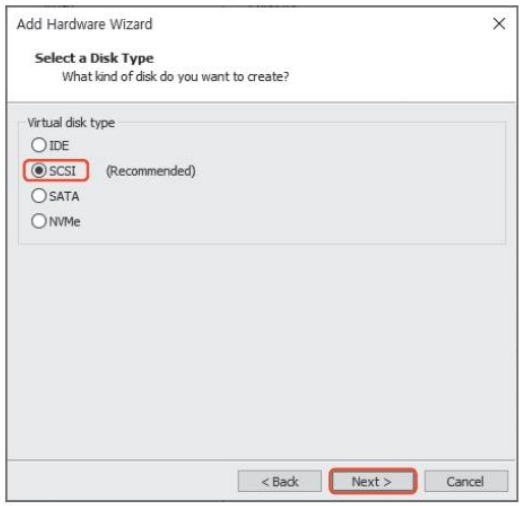
③ Select a Disk Type 창에서 디스크의 종류로 'SCSI'를 선택하고 [Next] 클릭

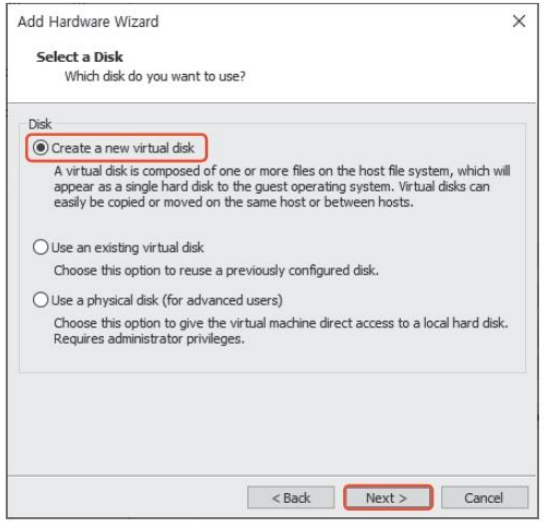
④ Select a Disk 창에서 'Create a new virtual disk'를 선택하고 [Next] 클릭

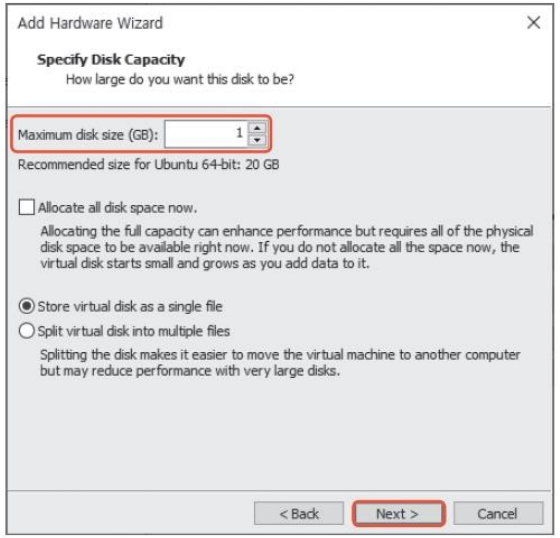
⑤ Specify Disk Capacity 창에서는 디스크의 용량을 설정: 1GB

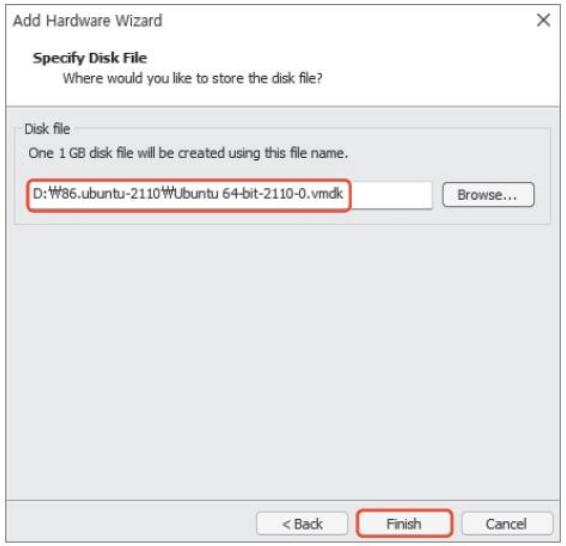
⑥ Specify Disk File 창에서 가상 디스크의 파일명을 지정

⑦ 디스크 파일이 생성되고 디스크 추가 작업이 완료









04. 디스크 추가 설치

■ 디스크 장치의 이름과 파티션 표시하기

- /dev/sd로 시작하는 이름을 사용
  - /dev/sda: 첫 번째 디스크
  - /dev/sdb: 두 번째 디스크
  - /dev/sdc: 세 번째 디스크
- 장치명이 /dev/sda라면 파티션의 이름을 다음과 같이 표시
  - /dev/sda: 첫 번째 디스크 전체를 의미하는 장치명
  - /dev/sda0: 디스크의 첫 번째 파티션
  - /dev/sda1: 디스크의 두 번째 파티션

04. 디스크 추가 설치

fdisk 명령

fdisk

- **기능** 디스크의 파티션 생성, 삭제, 보기 등 파티션을 관리한다.
- **형식** fdisk [옵션] [장치명]
- **옵션** -b 크기: 섹터 크기를 지정한다(512, 1024, 2048, 4096).
  -l: 파티션 테이블을 출력한다.
- **사용 예** fdisk /dev/sdb
  fdisk -l

04. 디스크 추가 설치

■ fdisk의 내부 명령

표 7-5 fdisk 명령의 내부 명령

명령	기능	명령	기능
a	부팅 파티션을 설정한다.	p	파티션 테이블을 출력한다.
b	BSD 디스크 라벨을 편집한다.	q	작업 내용을 저장하지 않고 종료한다.
c	도스 호환성을 설정한다.	s	새로운 빈 Sun 디스크 라벨을 생성한다.
d	파티션을 삭제한다.	t	파티션의 시스템 ID를 변경한다(파일 시스템 종류 변경).
l	사용 가능한 파티션의 종류를 출력한다.	u	항목 정보를 변경 · 출력한다.
m	도움말을 출력한다.	v	파티션 테이블을 검사한다.
n	새로운 파티션을 추가한다.	w	파티션 정보를 디스크에 저장하고 종료한다.
o	새로운 빈 DOS 파티션을 생성한다.	x	실린더 개수 변경 등 전문가를 위한 부가적 기능이다.

04. 디스크 추가 설치

■ 파티션 정보 보기: fdisk -l

```
user1@myubuntu:~$ sudo fdisk -l
(생략)
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Disk model: VMware Virtual S
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: gpt
Disk identifier: 7B7D9058-F142-4A97-B151-352BB25BB318
Device       Start      End  Sectors  Size Type
/dev/sda1     2048     4095     2048    1M BIOS boot
/dev/sda2     4096  1054719  1050624  513M EFI System
/dev/sda3  1054720 41940991 40886272 19.5G Linux filesystem
```

355p. 따라해보기 : 디스크 파티션 나누기

① **fdisk 명령을 실행: 장치명을 인자로 지정 (sudo fdisk /dev/sdb)**

② **새로운 파티션을 생성: n 입력**

③ **파티션의 종류를 선택**

- 기본(primary), 확장(extended)

④ **파티션 번호를 선택: 1**

⑤ **파티션의 크기 설정**

- 시작섹터: 기본값
- 마지막섹터: 크기 지정(+500MB)

⑥ **파티션 정보 확인: p**

⑦ **n을 입력하여 두번째 파티션 생성**

⑧ **w를 입력하여 파티션 설정정보 저장**

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

```
Select (default p): p
Partition number (1-4, default 1):
```

```
First sector (2048-2097151, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-2097151, default 2097151): +500M

Created a new partition 1 of type 'Linux' and of size 500 MiB.

Command (m for help):
```

04. 디스크 추가 설치

- 파티션을 만든 후 파일 시스템을 생성해야함
- 파일 시스템 생성 명령

mkfs

- **기능** 리눅스 파일 시스템을 만든다.
- **형식** mkfs [옵션] [장치명]
- **옵션** -t 종류: 파일 시스템의 종류를 지정한다(기본값은 ext2).
- **사용 예** mkfs /dev/sdb1
  mkfs -t ext4 /dev/sdb1

  - mkfs.ext2, mkfs.ext3, mkfs.ext4 명령도 제공

04. 디스크 추가 설치

■ 파일 시스템 생성 명령

- 설정 파일: /etc/mke2fs.conf

mke2fs

- **기능** 리눅스 개정판 확장 파일 시스템(ext2, ext3, ext4)을 만든다.
- **형식** mke2fs [옵션] [장치명]
- **옵션** -t 종류: 파일 시스템의 종류를 지정한다(기본값은 ext2).
  -b 블록 크기: 블록 크기를 바이트 수로 지정한다.
  -c: 배드 블록을 체크한다.
  -f 프래그먼트 크기: 프래그먼트 크기를 바이트 수로 지정한다.
  -i inode당 바이트 수: inode당 바이트 수를 지정한다(기본값은 4,096B).
  -m 예약 블록 퍼센트: 슈퍼유저에게 예약해둘 블록의 퍼센트를 지정한다(기본값은 5%).
- **사용 예** mke2fs /dev/sdb1
  mke2fs -t ext4 /dev/sdb1

362p. 따라해보기 : mkfs 명령으로 파일 시스템 생성하기

① mkfs 명령으로 /dev/sdb1 파티션에 ext2 파일 시스템 생성

② mkfs.ext3 명령으로 /dev/sdb2 파티션에 ext3 파일 시스템을 생성

```
user1@myubuntu:~$ sudo mkfs /dev/sdb1
mke2fs 1.46.3 (27-Jul-2021)
Creating filesystem with 128000 4k blocks and 128000 inodes
Filesystem UUID: b77be8f5-1299-4223-a831-58280caf302e
Superblock backups stored on blocks:
        32768, 98304

Allocating group tables: done
Writing inode tables: done
Writing superblocks and filesystem accounting information: done
```

```
user1@myubuntu:~$ sudo mkfs.ext3 -v /dev/sdb2
mke2fs 1.46.3 (27-Jul-2021)
fs_types for mke2fs.conf resolution: 'ext3', 'small'
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
128000 inodes, 128000 blocks
6400 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=134217728
4 block groups
32768 blocks per group, 32768 fragments per group
32000 inodes per group
Filesystem UUID: 83ee84c1-cdb2-45dc-a8e3-3e2f223fc936
Superblock backups stored on blocks:
        32768, 98304

Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

363p. 따라해보기 : mke2fs 명령으로 파일 시스템 생성하기

① **mke2fs 명령으로 /dev/sdc1 파티션에 ext3 파일 시스템을 생성**

② **mke2fs 명령으로 /dev/sdc2 파티션에 ext4 파일 시스템 생성**

- 블록 크기는 4,096B로 지정

```
user1@myubuntu:~$ sudo mke2fs -t ext4 -b 4096 /dev/sdc2
mke2fs 1.46.3 (27-Jul-2021)
Creating filesystem with 128000 4k blocks and 128000 inodes
Filesystem UUID: 0c2891e2-45c2-483e-9454-cddd4e4c23d7
Superblock backups stored on blocks:
        32768, 98304

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

```
user1@myubuntu:~$ sudo mke2fs -t ext3 /dev/sdc1
mke2fs 1.46.3 (27-Jul-2021)
Creating filesystem with 128000 4k blocks and 128000 inodes
Filesystem UUID: b41b4e0e-85ba-47e6-a854-e5f236bf0326
Superblock backups stored on blocks:
        32768, 98304

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

04. 디스크 추가 설치

- **마운트 포인트 준비하기**

```
user1@myubuntu:~$ sudo mkdir /mnt/hdd1
user1@myubuntu:~$ sudo mkdir /mnt/hdd2
user1@myubuntu:~$ sudo mkdir /mnt/hdd3
```

- **파일 시스템 마운트하기**

```
user1@myubuntu:~$ sudo mount /dev/sdb1 /mnt/hdd1
```

```
user1@myubuntu:~$ sudo mount -t ext3 /dev/sdb2 /mnt/hdd2
```

  - 마운트 결과

```
user1@myubuntu:~$ mount
(생략)
/dev/sdb1 on /mnt/hdd1 type ext2 (rw,relatime)
/dev/sdb2 on /mnt/hdd2 type ext3 (rw,relatime)
```

04. 디스크 추가 설치

■ 파일 시스템 사용하기

- 파일 복사하기

```
user1@myubuntu:~$ sudo cp /etc/hosts /mnt/hdd1
user1@myubuntu:~$ ls /mnt/hdd1
hosts  lost+found
```

- 이 상태에서 /mnt/hdd1의 마운트를 해제하면 /mnt/hdd1 디렉터리에는 어떤 파일이 있을까?

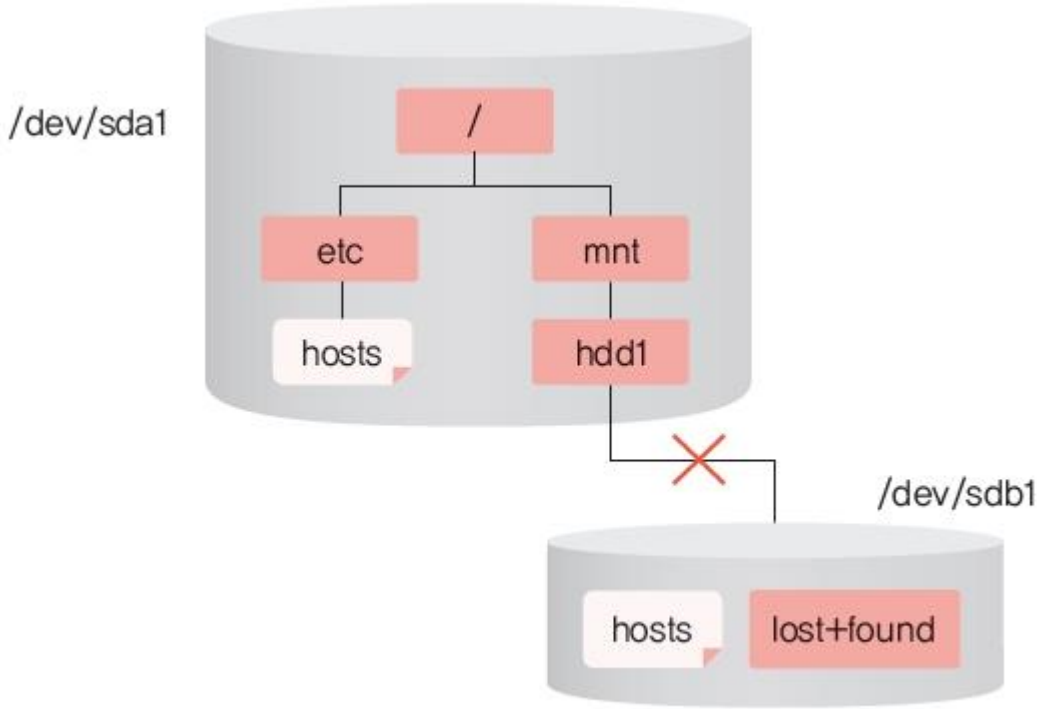


그림 7-13 디스크 마운트 해제와 파일의 관계

363p. 따라해보기 : mke2fs 명령으로 파일 시스템 생성하기

① **mke2fs 명령으로 /dev/sdc1 파티션에 ext3 파일 시스템을 생성**

② **mke2fs 명령으로 /dev/sdc2 파티션에 ext4 파일 시스템 생성**

- 블록 크기는 4,096B로 지정

```
user1@myubuntu:~$ sudo mke2fs -t ext4 -b 4096 /dev/sdc2
mke2fs 1.46.3 (27-Jul-2021)
Creating filesystem with 128000 4k blocks and 128000 inodes
Filesystem UUID: 0c2891e2-45c2-483e-9454-cddd4e4c23d7
Superblock backups stored on blocks:
        32768, 98304

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

```
user1@myubuntu:~$ sudo mke2fs -t ext3 /dev/sdc1
mke2fs 1.46.3 (27-Jul-2021)
Creating filesystem with 128000 4k blocks and 128000 inodes
Filesystem UUID: b41b4e0e-85ba-47e6-a854-e5f236bf0326
Superblock backups stored on blocks:
        32768, 98304

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

367p. 따라해보기 : 파일 시스템 마운트, 언마운트하기

① **/dev/sdb2의 파일 시스템을 언마운트**

② **/dev/sdc1의 파일 시스템을 /mnt/hdd1 디렉터리에 마운트**

③ **/dev/sdc2의 파일 시스템을 /mnt/hdd2 디렉터리에 마운트**

- 파일 시스템의 종류를 ext4로 지정

④ **/dev/sdc1과 /dev/sdc2의 파일 시스템이 제대로 마운트되었는지 확인**

⑤ **/dev/sdc2의 파일 시스템에 파일을 복사**

⑥ **dev/sdc1과 /dev/sdc2 파일 시스템의 마운트를 해제**

04. 디스크 추가 설치

■ 여러 디스크를 하나의 디스크로 사용하는 방법: LVM

- 독립적으로 구성된 디스크 파티션을 하나로 연결하여 한 파티션처럼 사용할 수 있게 함
- 관련 용어
  - PV physical volume(물리 볼륨): /dev/sdb1, /dev/sdb2 같은 실제 하드디스크의 파티션을 의미한다.
  - VG volume group(볼륨 그룹): 여러 개의 PV를 그룹으로 묶은 것을 말한다. 예를 들어 /dev/sdb1, /dev/sdb2가 GRP1이라는 그룹을 만든다면 GRP1을 VG라고 한다.
  - LV logical volume(논리 볼륨): VG를 다시 적절한 크기의 파티션으로 나눌 때 각 파티션을 LV라고 한다.
  - PE physical extent: PV가 가진 일정한 블록을 뜻한다.
  - LE logical extent: LV가 가진 일정한 블록을 뜻한다.

04. 디스크 추가 설치

LVM 예

- 10GB 용량의 물리적 파티션(PV) 세 개를 묶어서 30GB VG 하나를 만든 후 이를 다시 15GB LV 두 개로 분리

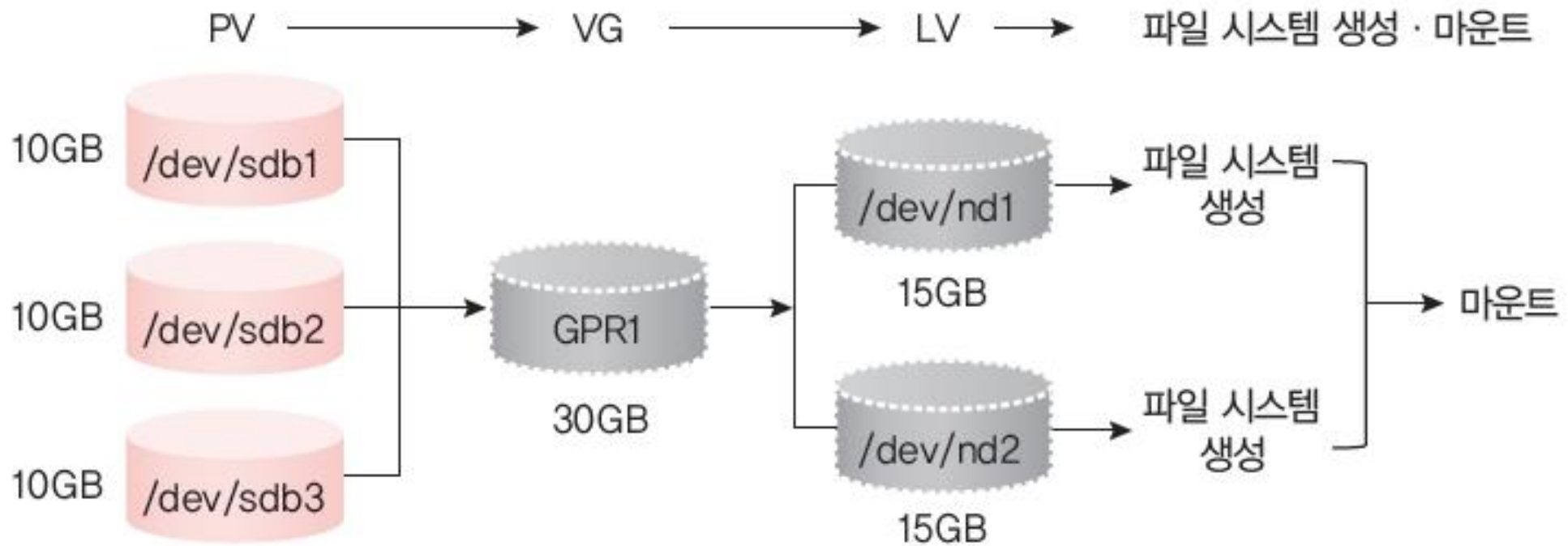


그림 7-14 LVM의 기본 개념

04. 디스크 추가 설치

LVM 관련 명령

표 7-6 LVM 관련 명령

구분	기능	명령
PV	PV 생성	pvcreate 파티션 이름
	PV 상태 확인	pvscan
VG	VG 생성	vgcreate VG명 파티션(PV)명1 파티션(PV)명2 …
	VG 활성화	vgchange –a y VG명
	VG 비활성화	vgchange –a n VG명
	VG 삭제	vgremove VG명
VG	VG 정보 확인	vgdisplay –v VG명
	VG에 PV 추가	vgextend VG명 PV명
	VG에서 PV 삭제	vgreduce VG명 PV명
	VG명 변경	vgrename 기존 VG명 새 VG명
LV	LV 생성	lvcreate –l PE 수 VG명 –n LV명
	LV 삭제	lvremove LV명
	LV 상태 확인	lvscan
	LV 용량 확대	lvextent –l +PE 수 LV명
	LV 용량 축소	lvextent –l –PE 수 LV명

04. 디스크 추가 설치

LVM 생성 과정

그림 7-15 LVM 생성 단계

370p. 따라해보기 : LVM 생성하기

/dev/sdb1, /dev/sdb2는 크기가 각각 500MB다. 이를 LVM으로 변환하고 1GB짜리 LV를 생성하여 마운트

① **lvm2 패키지를 먼저 설치: sudo apt install lvm2**
② **/dev/sdb1, /dev/sdb2의 파일 시스템 종류를 83(Linux)에서 8e(LinuxLVM)로 변경**
③ **/dev/sdb1, /dev/sdb2에 PV를 생성: sudo pvcreate /dev/sdb1**
④ **pvscan 명령으로 PV의 상태를 확인**
⑤ **두 PV를 통합하여 VG를 생성한다. VG의 이름은 grp1로 지정: sudo vgcreate grp1 /dev/sdb1 /dev/sdb2**
⑥ **생성된 VG grp1을 활성화: sudo vgchange -a y grp1**
⑦ **활성화된 VG grp1의 상태를 vgdisplay 명령으로 확인: sudo vgdisplay -v grp1**
⑧ **VG grp1에는 PE가 총 248개 있다. 이를 모두 합하여 하나의 LV를 생성**

: sudo lvcreate -l 248 grp1 -n mylvm1

⑨ **생성된 LV의 상태를 확인: sudo lvscan**
⑩ **LV mylvm1에 ext4 파일 시스템을 생성: 장치명: sudo mke2fs -t ext4 /dev/grp1/mylvm1**
⑪ **VG의 상태를 확인하여 LV의 정보가 수정되었는지 확인**
⑫ **LV를 /mnt/lvm 디렉터리에 마운트하고 파일을 복사: sudo mount /dev/grp1/mylvm1 /mnt/lvm**

05 디스크 관리

05. 디스크 관리

■ 파일 시스템별 디스크 사용량 확인하기: df

df

- **기능** 디스크의 남은 공간에 대한 정보를 출력한다.
- **형식** df [옵션] [파일 시스템]
- **옵션** -a: 모든 파일 시스템을 대상으로 디스크 사용량을 확인한다.
  -k: 디스크 사용량을 KB 단위로 출력한다.
  -m: 디스크 사용량을 MB 단위로 출력한다.
  -h: 디스크 사용량을 알기 쉬운 단위(GB, MB, KB 등)로 출력한다.
  -t 파일 시스템 종류: 지정한 파일 시스템 종류에 해당하는 디스크의 사용량을 출력한다.
  -T: 파일 시스템 종류를 출력한다.
- **사용 예** df
  df -h

05. 디스크 관리

- df 명령만 사용하는 경우

```
user1@myubuntu:~$ df
Filesystem              1K-blocks    Used Available Use% Mounted on
tmpfs                      398836    1908    396928   1% /run
/dev/sda3                19991152 8074116  10878496  43% /
tmpfs                     1994172       0   1994172   0% /dev/shm
tmpfs                        5120       4      5116   1% /run/l
tmpfs                        4096       0      4096   0% /sys/f
/dev/sda2                  524252    5340    518912   2% /boot/
```

- 파일 시스템 장치명
- 파일 시스템의 전체 용량
- 파일 시스템의 사용량
- 파일 시스템의 사용 가능한 남은 용량
- 퍼센트로 나타낸 사용량
- 마운트 포인트

05. 디스크 관리

- 파일 시스템 사용량을 이해하기 쉬운 단위로 표시하기: - h

```
user1@myubuntu:~$ df -h
Filesystem                Size  Used Avail Use% Mounted on
tmpfs                     390M  1.9M  388M   1% /run
/dev/sda3                  20G  7.8G   11G  43% /
tmpfs                     2.0G     0  2.0G   0% /dev/shm
tmpfs                     5.0M  4.0K  5.0M   1% /run/lock
tmpfs                     4.0M     0  4.0M   0% /sys/fs/cgroup
/dev/sda2                 512M  5.3M  507M   2% /boot/efi
```

05. 디스크 관리

- 파일 시스템의 종류 출력하기: -T

```
user1@myubuntu:~$ df -Th
Filesystem              Type   Size  Used Avail Use% Mounted on
tmpfs                   tmpfs  390M  1.9M  388M   1% /run
/dev/sda3               ext4    20G  7.8G   11G  43% /
tmpfs                   tmpfs  2.0G     0  2.0G   0% /dev/shm
tmpfs                   tmpfs  5.0M  4.0K  5.0M   1% /run/lock
tmpfs                   tmpfs  4.0M     0  4.0M   0% /sys/fs/cgroup
/dev/sda2               vfat   512M  5.3M  507M   2% /boot/efi
```

05. 디스크 관리

■ 디렉터리나 사용자별 디스크 사용량 확인하기: du

du

- **기능** 디스크의 사용 공간에 대한 정보를 출력한다.
- **형식** du [옵션] [디렉터리]
- **옵션** -s: 특정 디렉터리의 전체 사용량을 출력한다.
  -h: 디스크 사용량을 알기 쉬운 단위(GB, MB, KB 등)로 출력한다.
- **사용 예** du
  du -s ~user1

05. 디스크 관리

- du 명령만 사용하는 경우: 현재 디렉터리의 디스크 사용량 출력

```
user1@myubuntu:~$ pwd
/home/user1
user1@myubuntu:~$ du
4       ./temp/mid/han
8       ./temp/mid
12      ./temp
20      ./linux_ex/ch2/temp
44      ./linux_ex/ch2
4       ./linux_ex/ch5/temp
(생략)
664     ./.local/share
668     ./.local
6228    .
```

05. 디스크 관리

- 전체 디스크 사용량 출력하기: 지정한 디렉터리의 전체 사용량만 출력

```
user1@myubuntu:~$ du -s
6228    .
user1@myubuntu:~$ sudo du -s /etc
15312   /etc
```

- 특정 사용자의 디스크 사용량 출력: 사용자의 홈 디렉터리를 지정

```
user1@myubuntu:~$ du -sh ~user1
6.1M    /home/user1
```

380p. 따라해보기 : 디스크 사용량 확인하기

① 파일 시스템의 디스크 사용량을 MB 단위로 출력: df -m
② ext4 파일 시스템의 디스크 사용량을 출력: df -t ext4
③ 전체 파일 시스템의 디스크 사용량을 출력: du -a
④ /usr 디렉터리가 사용하고 있는 디스크 사용량을 출력: sudo du -sh /usr

05. 디스크 관리

■ 파일 시스템 검사하기: fsck

fsck

- **기능** 리눅스의 파일 시스템을 점검한다.
- **형식** fsck [옵션] [장치명]
- **옵션** -f: 강제로 점검한다.
  -b 슈퍼블록: 슈퍼블록으로 지정한 백업 슈퍼블록을 사용한다.
  -y: 모든 질문에 yes로 대답하도록 한다.
  -a: 파일 시스템 검사에서 문제를 발견했을 때 자동으로 복구한다.
- **사용 예** fsck /dev/sdb1
  fsck -f /dev/sdb1

▪ 파일 시스템 종류별로 fsck.ext2, fsck.ext3, fsck.ext4 명령도 제공

05. 디스크 관리

- 일반적인 파일 시스템 검사: sudo fsck /dev/sdd1

```
user1@myubuntu:~$ sudo fsck /dev/sdd1
fsck from util-linux 2.36.1
e2fsck 1.46.3 (27-Jul-2021)
/dev/sdd1: clean, 11/76800 files, 2452/76800 blocks
```

05. 디스크 관리

- 파일 시스템 강제 검사: sudo fsck -f /dev/sdd1

```
user1@myubuntu:~$ sudo fsck -f /dev/sdd1
fsck from util-linux 2.36.1
e2fsck 1.46.3 (27-Jul-2021)
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
/dev/sdd1: 11/76800 files (0.0% non-contiguous), 2452/76800 blocks
```

05. 디스크 관리

- 파일 시스템 종류를 지정한 검사

```
user1@myubuntu:~$ sudo fsck.ext4 /dev/sdd3
e2fsck 1.46.3 (27-Jul-2021)
/dev/sdd3: clean, 11/102400 files, 7463/102400 blocks
```

05. 디스크 관리

■ 파일 시스템 검사하기: e2fsck

e2fsck

- **기능** 리눅스의 확장 파일 시스템(ext2, ext3, ext4)을 점검한다.
- **형식** e2fsck [옵션] [장치명]
- **옵션** -f: 강제로 점검한다.
  -b 슈퍼블록: 슈퍼블록으로 지정한 백업 슈퍼블록을 사용한다.
  -y: 모든 질문에 yes로 대답하도록 한다.
  -j ext3/ext4: ext3나 ext4 파일 시스템을 검사할 때 지정한다.
- **사용 예** e2fsck /dev/sdb1
  e2fsck -f /dev/sdb1

05. 디스크 관리

- 일반적인 파일 시스템 검사

```
user1@myubuntu:~$ sudo e2fsck /dev/sdd1
e2fsck 1.46.3 (27-Jul-2021)
/dev/sdd1: clean, 11/76800 files, 2452/76800 blocks
```

05. 디스크 관리

- 파일 시스템 강제 검사

```
user1@myubuntu:~$ sudo e2fsck -f /dev/sdd1
e2fsck 1.46.3 (27-Jul-2021)
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
/dev/sdd1: 11/76800 files (0.0% non-contiguous), 2452/76800 blocks
```

05. 디스크 관리

■ 배드 블록 검사하기: badblocks

badblocks

- **기능** 장치의 배드 블록을 검색한다.
- **형식** badblocks [옵션] [장치명]
- **옵션** -v: 검색 결과를 자세하게 출력한다.
  -o 출력 파일: 검색한 배드 블록 목록을 지정한 출력 파일에 저장한다.
- **사용 예** badblocks -v /dev/sdb1
  badblocks -v -o bad.out /dev/sdb1

05. 디스크 관리

- 배드 블록 검색하기

```
user1@myubuntu:~$ sudo badblocks -v /dev/sdd1
Checking blocks 0 to 307199
Checking for bad blocks (read-only test): done


Pass completed, 0 bad blocks found. (0/0/0 errors)
```

05. 디스크 관리

- 검색 결과를 파일로 저장하기: -o

```
user1@myubuntu:~$ sudo badblocks -v -o bad.out /dev/sdd1
Checking blocks 0 to 307199
Checking for bad blocks (read-only test): done


Pass completed, 0 bad blocks found. (0/0/0 errors)
user1@myubuntu:~$ cat bad.out
```

05. 디스크 관리

■ 백업 슈퍼블록을 이용하여 파일 시스템 복구하기

- 백업 슈퍼블록의 위치 파악하기: dumpe2fs

dumpe2fs

- **기능** 파일 시스템의 정보를 출력한다.
- **형식** dumpe2fs [장치명]
- **사용 예** dumpe2fs /dev/sdd1

```
user1@myubuntu:~$ sudo dumpe2fs /dev/sdd1 | grep superblock
dumpe2fs 1.46.3 (27-Jul-2021)
  Primary superblock at 0, Group descriptors at 1-1
  Backup superblock at 32768, Group descriptors at 32769-32769
```

05. 디스크 관리

- 파일 시스템의 블록 복사 명령

dd

- **기능** 지정한 블록 크기만큼 파일을 복사한다.
- **형식** dd [if=파일] [of=파일] [bs=바이트 수] [count=블록 수]
  - if=파일: 표준 입력 대신 지정한 파일에서 읽어온다.
  - of=파일: 표준 출력 대신 지정한 파일로 복사한다.
  - bs=바이트 수: 한 번에 읽어오고 기록할 바이트 수다.
  - count=블록 수: 블록 수만큼만 복사한다.
- **사용 예** dd if=/dev/zero of=/dev/sdd1 bs=4096 count=20

05. 디스크 관리

- 파일 시스템 복구하기

① 파일 시스템의 기본 슈퍼블록을 dd 명령으로 삭제: 파일 시스템의 앞부분 20블록을 0 값으로 채우기

```
user1@myubuntu:~$ sudo dd if=/dev/zero of=/dev/sdd1 bs=4096 count=20
20+0 records in
20+0 records out
81920 bytes (82 kB, 80 KiB) copied, 0.00865865 s, 9.5 MB/s#
```

② /dev/sdd1 파일 시스템을 마운트하면 오류 메시지가 출력

```
user1@myubuntu:~$ sudo mount /dev/sdd1 /mnt/hdd1
mount: /mnt/hdd1: wrong fs type, bad option, bad superblock on /dev/sdd1, missing
codepage or helper program, or other error.
```

③ dumpe2fs로 확인한 백업 슈퍼블록을 이용하여 /dev/sdd1 파일 시스템을 복구

- sudo e2fsck -b 32768 -y /dev/sdd1

05. 디스크 관리

```
user1@myubuntu:~$ sudo e2fsck -b 32768 -y /dev/sdd1
e2fsck 1.46.3 (27-Jul-2021)
/dev/sdd1 was not cleanly unmounted, check forced.
Resize inode not valid.  Recreate? yes

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
Free blocks count wrong for group #0 (31939, counted=31940).
Fix? yes

Free blocks count wrong (74347, counted=74348).
Fix? yes



/dev/sdd1: ***** FILE SYSTEM WAS MODIFIED *****
/dev/sdd1: 11/76800 files (0.0% non-contiguous), 2452/76800 blocks
```

Thank You !

한빛아카데미 Hanbit Academy, Inc.