

Chapter 05

파일 시스템 및 디스크 관리



학습목표

- 리눅스의 디렉토리 구조를 설명할 수 있다.
- 파일 시스템이 무엇이고 리눅스에서 사용하는 파일시스템의 종류를 설명할 수 있다.
- 리눅스의 디스크관리 체계를 설명할 수 있다.
- 저장 매체를 추가하여 사용할 수 있다.



학습내용

- ❖ 리눅스 파일 시스템
 - 리눅스 디렉토리 구조
 - 리눅스 파일과 디렉터리 이름 규칙
 - 파일 시스템의 종류
- ❖ 리눅스 디스크 관리
 - 저장 매체 인터페이스
 - 하드디스크 추가하기
 - 디스크 관리 명령어



리눅스 디렉토리 구조

디렉토리 계층 구조

- 리눅스에서는 파일을 효율적으로 관리하기 위해 디렉터리를 계층적으로 구성 -> 트리(tree) 구조
- 모든 디렉터리의 출발점은 루트(root, 뿌리) 디렉터리이며, /(빗금)으로 표시

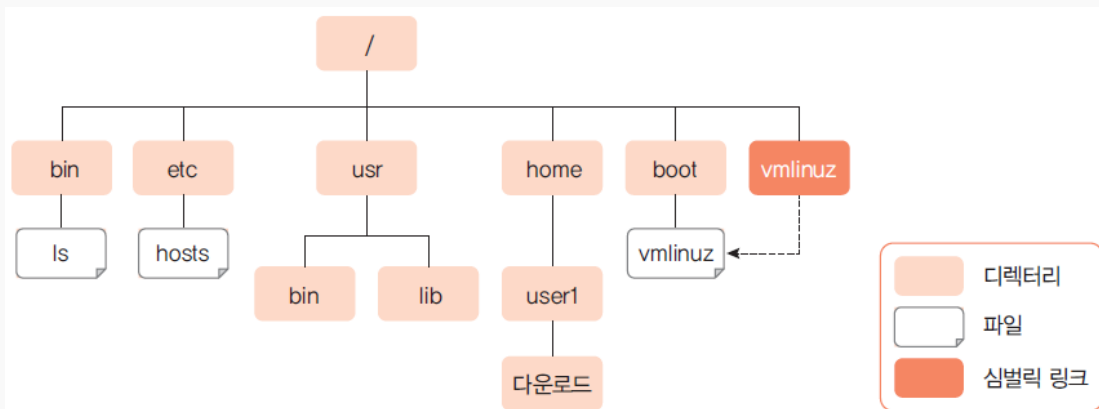


그림 2-2 디렉터리 계층 구조의 예



리눅스 디렉토리 구조

디렉토리 계층 구조

- 하위 디렉터리(서브 디렉터리): 디렉터리 아래에 있는 디렉터리 (bin, etc, usr, home, boot)
- 상위 디렉터리(부모 디렉터리): ‘..’으로 표시
- 루트 디렉터리를 제외하고 모든 디렉터리에는 부모 디렉터리가 있음



리눅스 디렉토리 구조

루트 디렉터리의 서브 디렉터리

```
root@server:~# cd /
root@server:/# ls -F
bin/      etc/      lib/      mnt/      run/      swapfile  var/
boot/     home/     lib64/     opt/      sbin/     sys/      vmlinuz@
cdrom/    initrd.img@  lost+found/  proc/    snap/     tmp/
dev/      initrd.img.old@ media/      root/    srv/      usr/
```

- / : 해당 파일이 디렉토리임을 표시
- @ : 심볼릭 링크

작업 디렉토리

- 현재 사용 중인 디렉터를 작업 디렉터리(working directory) 또는 현재 디렉터리(current directory)라고 함
- 현재 디렉터리는 ‘.’ 기호로 표시
- 현재 디렉터리의 위치는 pwd 명령으로 확인



리눅스 디렉토리 구조

홈 디렉터리

- 각 사용자에게 할당된 디렉터리로 처음 사용자 계정을 만들 때 지정
- 사용자는 자신의 홈 디렉터리 아래에 파일이나 서브 디렉터를 생성하며 작업 가능
- 홈 디렉터리는 '~' 기호로 표시 : ~user1



리눅스 디렉토리 구조

디렉토리의 주요 기능

표 2-1 디렉터리의 주요 기능

디렉터리	기능
dev	장치 파일이 담긴 디렉터리이다.
home	사용자 홈 디렉터리가 생성되는 디렉터리이다.
media	CD-ROM이나 USB 같은 외부 장치를 연결(마운트라고 함)하는 디렉터리이다.
opt	추가 패키지가 설치되는 디렉터리이다.
root	root 계정의 홈 디렉터리이다. 루트(/) 디렉터리와 다른 것이므로 혼동하지 않도록 한다.
sys	리눅스 커널과 관련된 파일이 있는 디렉터리이다.
usr	기본 실행 파일과 라이브러리 파일, 헤더 파일 등 많은 파일이 있다. 참고로 usr는 'Unix System Resource'의 약자이다.
boot	부팅에 필요한 커널 파일을 가지고 있다.
etc	리눅스 설정을 위한 각종 파일을 가지고 있다.
lost+found	파일 시스템에 문제가 발생하여 복구할 경우, 문제가 되는 파일이 저장되는 디렉터리로 보통은 비어 있다.
mnt	파일 시스템을 임시로 마운트하는 디렉터리이다.
proc	프로세스 정보 등 커널 관련 정보가 저장되는 디렉터리이다.
run	실행 중인 서비스와 관련된 파일이 저장된다.
srv	FTP나 Web 등 시스템에서 제공하는 서비스의 데이터가 저장된다.
tmp	시스템 사용 중에 발생하는 임시 데이터가 저장된다. 이 디렉터리에 있는 파일은 재시작하면 모두 삭제된다.
var	시스템 운영 중에 발생하는 데이터나 로그 등 내용이 자주 바뀌는 파일이 주로 저장된다.



리눅스 디렉토리 용어

경로명

- 파일 시스템에서 디렉터리 계층 구조에 있는 특정 파일이나 디렉터리의 위치 표시
- 경로명에서 각 경로를 구분하는 구분자로 /을 사용
- 경로명에서 가장 앞에 있는 /은 루트 디렉터리를 뜻하지만 경로명 중간에 있는 /은 구분자
- 예: bin/ls에서 맨 앞의 /은 루트 디렉터리를 의미하고, 중간에 있는 /은 디렉터리 이름과 파일명을 구분하는 구분자

절대 경로명

- 항상 루트 디렉터리인 /로 시작
- 루트 디렉터리부터 시작하여 특정 파일이나 디렉터리의 위치까지 이동하면서 거치게 되는 모든 중간 디렉터리의 이름을 표시
- 특정 위치를 가리키는 절대 경로명은 항상 동일



리눅스 디렉토리 용어

상대 경로명

- 현재 디렉터리를 기준으로 시작
- / 이외의 문자로 시작
- 현재 디렉터리를 기준으로 서브 디렉터리로 내려가면 그냥 서브 디렉터리명을 추가
- 현재 디렉터리를 기준으로 상위 디렉터리로 가려면 ..(마침표 두 개)를 추가
- 상대 경로명은 현재 디렉터리가 어디냐에 따라 달라짐



리눅스 파일과 디렉터리 이름 규칙

파일과 디렉터리 이름 규칙

- 파일과 디렉터리 이름에는 /을 사용할 수 없다. /은 경로명에서 구분자로 사용하기 때문이다.
- 파일과 디렉터리의 이름에는 알파벳, 숫자, 붙임표(-), 밑줄(_), 마침표(.)만 사용한다.
- 파일과 디렉터리의 이름에는 공백문자, *, |, ", ', @, #, \$, %, ^, & 등을 사용하면 안 된다.
- 파일과 디렉터리 이름의 영문은 대문자와 소문자를 구별하여 다른 글자로 취급한다.
- 파일과 디렉터리의 이름이 .(마침표)로 시작하면 숨김 파일로 간주한다.

사용할 수 없는 파일 명

- &game, my home, myhome/, /test, bad/game



파일 시스템의 종류

파일 시스템

- 파일과 디렉터리의 집합을 구조적으로 관리하는 체계
- 어떤 구조를 구성하여 파일이나 디렉터리를 관리하느냐에 따라 다양한 형식의 파일 시스템이 존재
- 윈도우 운영체제의 파일 시스템은 NTFS 와 FAT 파일 시스템
- NTFS
 - ✓ New Technology File System
 - ✓ 윈도우 NT 계열 운영체제의 파일 시스템으로 윈도우 2000이후 모든 윈도우에서 사용
- FAT
 - ✓ File Allocation Table
 - ✓ MS-DOS시절부터 사용했던 파일 시스템
 - ✓ USB 메모리, 메모리 카드 등은 지금도 모두 FAT파일 시스템으로 되어 있음.
 - ✓ 안정성이 떨어지고 낭비가 많아 NTFS로 대체됨



파일 시스템의 종류

리눅스의 파일 시스템

- 리눅스는 윈도우에 비해 훨씬 많은 파일 시스템을 인식 할 수 있음.
- 리눅스 고유의 파일 시스템으로 ext1, ext2, ext3, ext4등이 있음.
- ext1
 - ✓ Extended File System'의 약자로 1992년 4월 리눅스 0.96c에 포함되어 발표
 - ✓ 파일 시스템의 최대 크기는 2GB, 파일 이름의 길이는 255바이트까지 지원
 - ✓ inode 수정과 데이터의 수정 시간 지원이 안 되고, 파일 시스템이 복잡해지고 파편화되는 문제
- ext2
 - ✓ ext1 파일 시스템이 가지고 있던 문제를 해결하고, 1993년 1월 발표
 - ✓ ext2는 ext3 파일 시스템이 도입되기 전까지 사실상 리눅스의 표준 파일 시스템으로 사용
 - ✓ 이론적으로 32TB까지 가능



파일 시스템의 종류

리눅스의 파일 시스템

- ext3
 - ✓ ext3는 ext2를 기반으로 개발되어 호환이 가능하며 2001년 11월 공개
 - ✓ ext3의 가장 큰 장점은 저널링(journaling) 기능을 도입 복구기능 강화
 - ✓ 파일 시스템의 최대 크기는 블록의 크기에 따라 2~32TB까지 지원
- ext4
 - ✓ ext4 파일 시스템은 1EB(엑사바이트, 1EB=1,024×1,024TB) 이상의 볼륨과 16TB 이상의 파일을 지원
 - ✓ ext2 및 ext3와 호환성을 유지하며 2008년 12월 발표
- Journaling 기능
 - ✓ 갑작스런 정전등으로 데이터가 날아가는 걸 방지하는 기능
 - ✓ 중요한 기업형 서버등으로 사용될 때 반드시 필요한 기능
 - ✓ 근래에는 임베디드 시스템과 같은 모바일 제품도 journaling기능이 있는 파일시스템 채용



파일 시스템의 종류

리눅스가 지원하는 파일 시스템

- 리눅스는 현존하는 거의 모든 파일시스템을 인식할 수 있음.
 - ✓ 리눅스에서 사용하던 하드디스크를 윈도우PC에서는 인식하지 못하지만 윈도우PC에서 사용하던 하드디스크는 리눅스에서 인식함.

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 시스템으로 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 시스템/V를 지원하기 위한 파일 시스템이다.
hfs	맥 컴퓨터의 hfs 파일 시스템을 지원하기 위한 파일 시스템이다.



파일 시스템의 종류

특수 용도의 가상 파일 시스템

표 7-2 리눅스의 가상 파일 시스템

파일 시스템	기능
swap	<ul style="list-style-type: none">스왑 영역을 관리하기 위한 스왑 파일 시스템이다.우분투 17.04부터는 스왑 파일 시스템 대신 스왑 파일을 사용한다.
tmpfs	<ul style="list-style-type: none">temporary file system으로 메모리에 임시 파일을 저장하기 위한 파일 시스템이며, 시스템이 재시작 할 때마다 기존 내용이 없어진다./run 디렉터리를 예로 들 수 있다.
proc	<ul style="list-style-type: none">proc 파일 시스템으로 /proc 디렉터리이다.커널의 현재 상태를 나타내는 파일을 가지고 있다.
ramfs	<ul style="list-style-type: none">램디스크를 지원하는 파일 시스템이다.
rootfs	<ul style="list-style-type: none">root file system으로 / 디렉터리이다.시스템 초기화 및 관리에 필요한 내용을 관리한다.



파일 시스템의 종류

지원하는 모든 파일 시스템 확인

- /proc/filesystems 파일을 확인하면 현재 커널이 지원하는 파일 시스템의 종류를 알려줌.
- \$ cat /proc/filesystems로 확인
- 앞에 nodev라는 용어는 해당 파일 시스템에 디스크가 연결되어 있지 않다는 의미로 가상 파일시스템을 의미함

```
root@server:/# cat /proc/filesystems
nodev    sysfs
nodev    rootfs
nodev    ramfs
nodev    bdev
nodev    proc
nodev    cpuset
nodev    cgroup
nodev    cgroup2
nodev    tmpfs
nodev    devtmpfs
nodev    configfs
nodev    debugfs
nodev    tracefs
nodev    securityfs
nodev    sockfs
nodev    dax
nodev    bpf
```

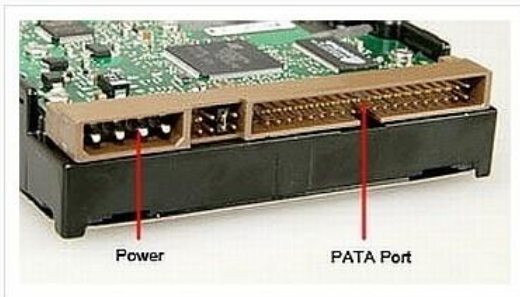
```
nodev    pipefs
nodev    hugetlbfs
nodev    devpts
          ext3
          ext2
          ext4
          squashfs
          vfat
nodev    ecryptfs
nodev    fuseblk
nodev    fuse
nodev    fusectl
nodev    pstore
nodev    mqueue
nodev    autofs
          iso9660
root@server:/#
```




저장 매체 인터페이스

하드 디스크 인터페이스

- 일반적으로 많이 사용하는 저장 매체는 하드 디스크, USB 메모리, CD-ROM 등이 있다.
- 이와 같은 저장 매체는 읽고 쓰는 속도가 중요하기 때문에 성능 향상을 위한 노력으로 인터페이스가 계속 발전하였다.
- 하드 디스크의 인터페이스는 IDE, SATA, SCSI 방식등이 있음.
- 리눅스의 디스크 관리 명령어들은 이들 인터페이스와 관련이 있어 여기서 이들 용어를 정리
- IDE (Integrated Drive Electronics)
 - ✓ 가장 오래된 규격으로 포트는 40개의 핀으로 구성
 - ✓ 아이디어라고 읽음
 - ✓ 데이터를 병렬로 전송한다는 뜻에서 PATA(Parallel Advanced Technology Attachment) 인터페이스로도 불림
 - ✓ 초당 133.3MB 전송 속도



리눅스 디스크 관리



저장 매체 인터페이스

하드 디스크 인터페이스

리눅스는 CD-ROM과 같은 광학 저장 매체를 사용할 때 SATA방식으로 인식

- SATA(Serial Advanced Technology Attachment)

- ✓ 현재 가장 많이 사용되는 인터페이스로 IDE보다 성능이 많이 향상됨.
- ✓ 싸타라고 읽음
- ✓ SATA1은 초당 150MB, SATA2는 초당 300MB속도
- ✓ SATA2는 USB처럼 허브를 통해 한 포트에 여러 개의 하드를 연결할 수 있음.

- SCSI(Small Computer System Interface)

- ✓ 서버나 워크스테이션 등에 쓰이는 고속 인터페이스
- ✓ 스카시라고 읽음
- ✓ 무엇보다 안정성이 높은 것이 최대의 장점이지만 고가
- ✓ 최신 규격인 울트라 320은 초당 320MB





하드디스크 추가하기

가상머신의 저장 매체 연결 상태

- Server 가상 머신에 하드디스크를 1개를 추가하는 실습을 진행하여 리눅스 디스크 관리의 기본을 이해.
- 그림과 같이 Vmware는 SATA방식 슬롯 4개(SATA0~3), SCSI방식 슬롯 4개(SCSI0~3)를 제공 (슬롯은 꽂을 수 있는 커넥터를 말함)
- 한 개의 슬롯인 SATA0에는 총 30개의 SATA장치 연결 가능 (SATA0:0~29)
- 그래서 SATA장치는 $4 \times 30 = 120$ 개 장착 가능
- SATA방식의 디바이스 파일은 `/dev/sr0`

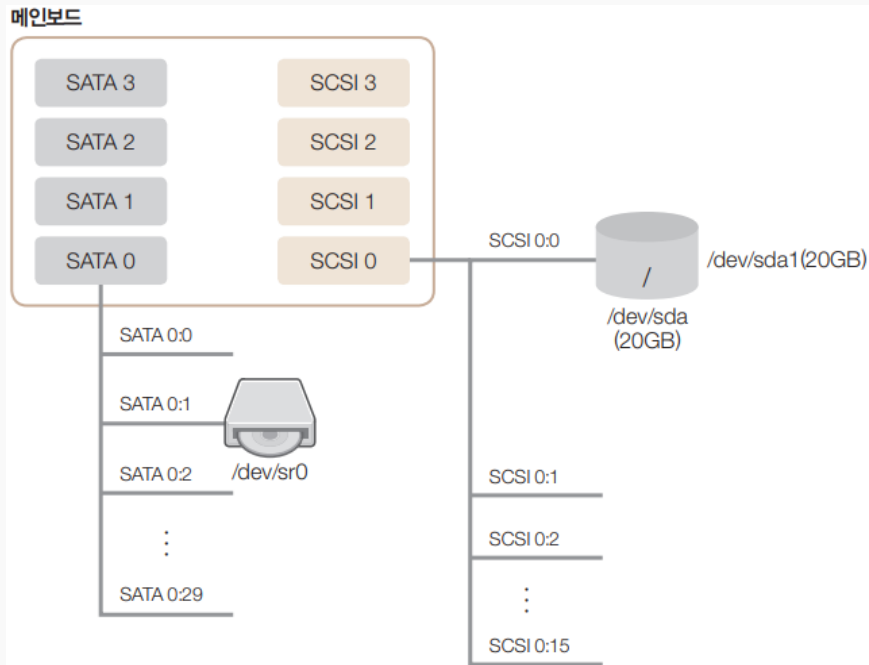
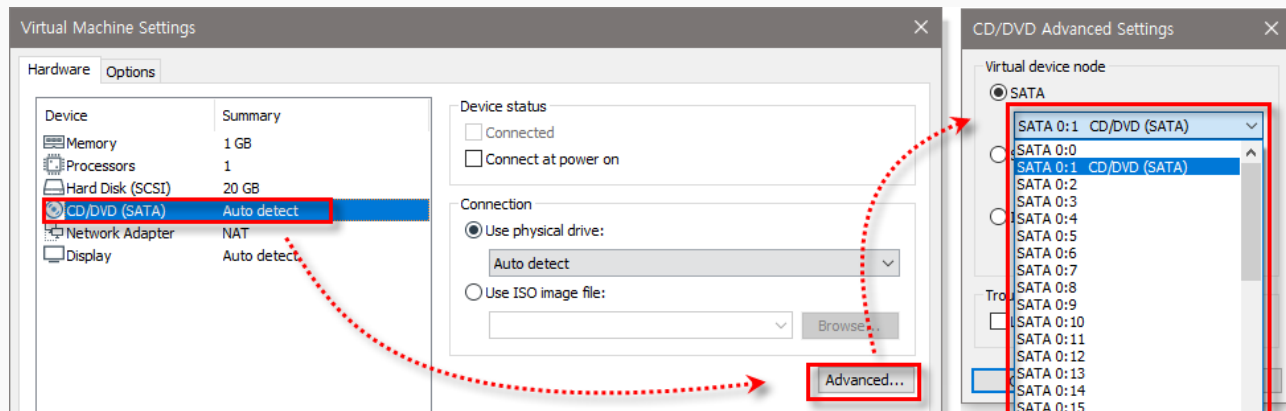


그림 9-1 Server 가상머신의 디스크 구성



하드디스크 추가하기

가상머신의 저장 매체 연결 상태

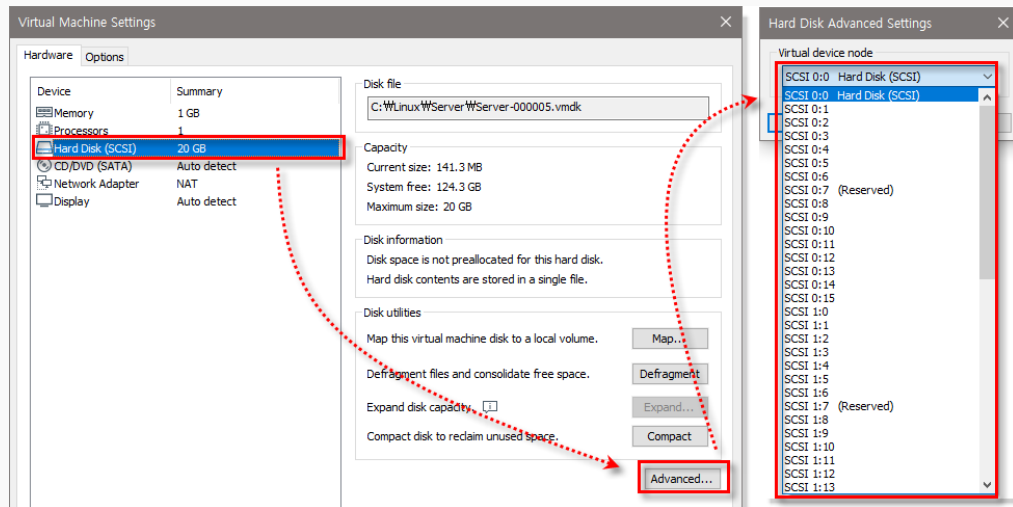


- SATA 장치는 주로 SATA 0:0, SATA 0:1, SATA 0:2 ... 로 표기
- VMware에는 기본적으로 SATA 0:1에 CD/DVD 장치가 장착됨
- SATA 장치(주로 디스크) 추가는 비어 있는 나머지 119개의 장치에 장착 가능
- [Virtual Machine Settings]를 클릭하면 SATA 장치 확인 가능



하드디스크 추가하기

가상머신의 저장 매체 연결 상태



- 하드디스크의 인터페이스인 SCSI0에는 총 15개의 SCSI장치 연결 가능 (SCSI:0~15, SCSI, 7번인 SCSI 0:7은 Reserved로 제외)
- SCSI 1, 2, 3번 슬롯도 각각 사용 가능하므로 총 60개(4×15)의 디스크 장착 가능



하드디스크 추가하기

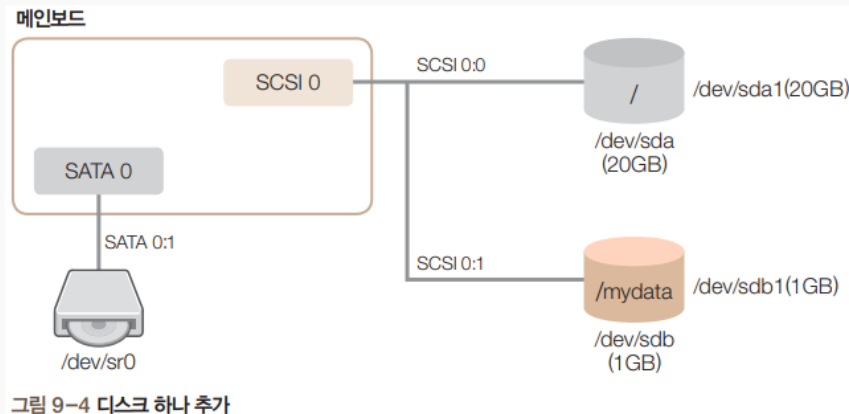
저장 매체 디바이스 파일의 규칙

- 리눅스는 외부 디바이스를 다룰 때 장치를 파일로 추상화하여 마치 파일처럼 다룸.
- /dev 디렉토리에 있는 파일들은 실제 파일이 아닌 디바이스 파일임.
- 하드디스크도 디바이스 파일로 다루게 되면 다음과 같은 규칙을 가짐.
- 처음 장착된 SCSI 디스크를 /dev/sda, 추가로 장착된 SCSI 디스크를 /dev/sdb, /dev/sdc, /dev/sdd, ...로 부름
- 하나의 디스크를 여러 개의 파티션으로 나눌 수 있음. 윈도우에서 하나의 디스크를 C:, D:등으로 나누는 것과 같음.
- /dev/sda 장치를 파티션으로 나누면 이 파티션들은 차례대로 1, 2, 3, 4를 붙여서 /dev/sda1, /dev/sda2, /dev/sda3, /dev/sda4라고 부름
- 그림 9-1을 보면 /dev/sda 장치에 파티션 하나를 설정했기 때문에 /dev/sda1 파티션만 있음



하드디스크 추가하기

1 개의 디스크 추가하기



- 추가한 디스크의 이름은 `/dev/sdb`
- 파티션의 논리적 이름은 `/dev/sdb1`
- `/mydata`라는 디렉터리를 만들고 이 디렉터리에 `/dev/sdb1` 를 마운트



하드디스크 추가하기

마운트(mount)란?

- 마운트는 하드디스크나 CD-ROM같은 저장 매체의 파일 시스템을 인식하여 접근이 가능하도록 임의의 디렉토리에 연결하는 것을 말함.
- 윈도우OS는 자동으로 마운트되기 때문에 느끼지 못하지만 리눅스는 사용자가 직접 마운트해 주어야 함 (자동 마운트 기능 제외)
- 예) 앞에서 새로 추가한 하드디스크의 첫 번째 파티션은 /dev/sdb1에 잡히는데 이 파티션의 내용을 확인하려면 /mydata와 같은 임의의 디렉터리를 만들고 마운트 시켜야 함.
- /mydata와 같은 디렉토리를 마운트 디렉토리 또는 마운트 포인트라고 함.
- 마운트 명령어 : `$ mount [-t 파일시스템타입] [디바이스파일] [디렉토리명]`
- 직접 마운트 하는 것은 Root만이 가능함
- 현재 마운트되어 있는 정보는 누구나 열람 가능

리눅스 디스크 관리



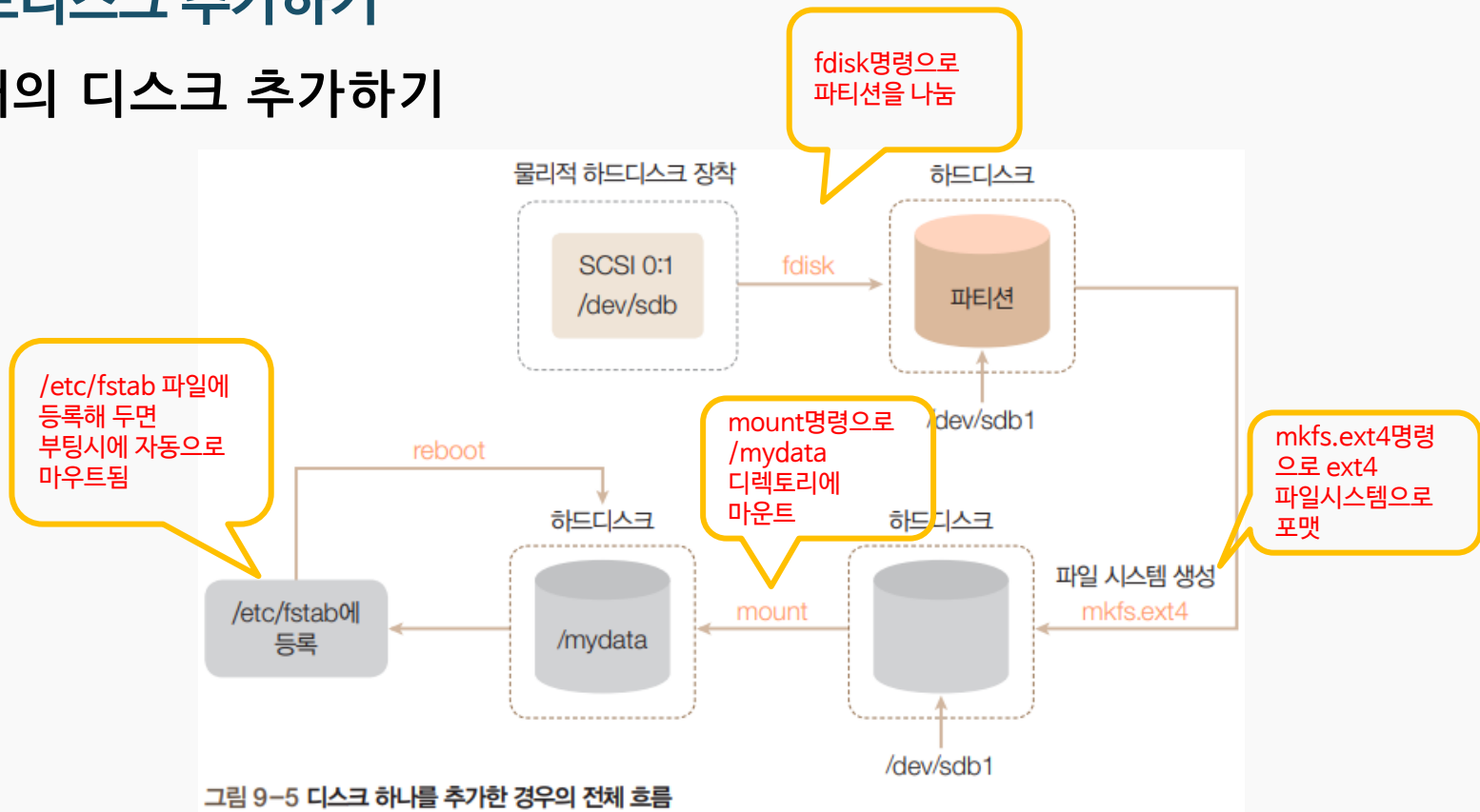
하드디스크 추가하기

마운트(mount)란?

- 명령어 df는 현재 시스템에 연결된 디스크 정보를 알려줌. df -h 를 실행
- /dev/sda1이 우분투 리눅스를 설치할 때 잡았던 용량 20GB의 하드디스크이고 Mounted On의 / 가 마운트 디렉토리
- /dev/sr0가 우분투 리눅스 설치 CD파일인 ubuntu-18.04.2-desktop-amd64.iso를 말하며 /media/root/Ubuntu 18.04.2 LTS amd64가 마운트 디렉토리

```
root@server:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            955M   0    955M   0% /dev
tmpfs           197M  1.5M  196M   1% /run
/dev/sda1       20G   6.2G   13G   33% /
tmpfs           985M   0    985M   0% /dev/shm
tmpfs           5.0M  4.0K   5.0M   1% /run/lock
tmpfs           985M   0    985M   0% /sys/fs/cgroup
/dev/loop0       4.3M  4.3M   0 100% /snap/gnome-calculator/544
/dev/loop1      141M  141M   0 100% /snap/gnome-3-26-1604/98
/dev/loop2       55M   55M   0 100% /snap/core18/1668
/dev/loop3      141M  141M   0 100% /snap/gnome-3-26-1604/74
/dev/loop4       35M   35M   0 100% /snap/gtk-common-themes/818
/dev/loop5       91M   91M   0 100% /snap/core/6350
/dev/loop6       13M   13M   0 100% /snap/gnome-characters/139
/dev/loop7       15M   15M   0 100% /snap/gnome-logs/45
/dev/loop9       2.3M  2.3M   0 100% /snap/gnome-calculator/260
/dev/loop8       3.8M  3.8M   0 100% /snap/gnome-system-monitor/57
/dev/loop10      3.8M  3.8M   0 100% /snap/gnome-system-monitor/127
/dev/loop11      1.0M  1.0M   0 100% /snap/gnome-logs/81
/dev/loop12      15M   15M   0 100% /snap/gnome-characters/399
/dev/loop13     161M  161M   0 100% /snap/gnome-3-28-1804/116
/dev/loop14      45M   45M   0 100% /snap/gtk-common-themes/1440
/dev/loop15      92M   92M   0 100% /snap/core/8592
tmpfs           197M  20K   197M   1% /run/user/0
/dev/sr0        1.9G  1.9G   0 100% /media/root/Ubuntu 18.04.2 LTS amd64
root@server:~#
```

1개의 디스크 추가하기

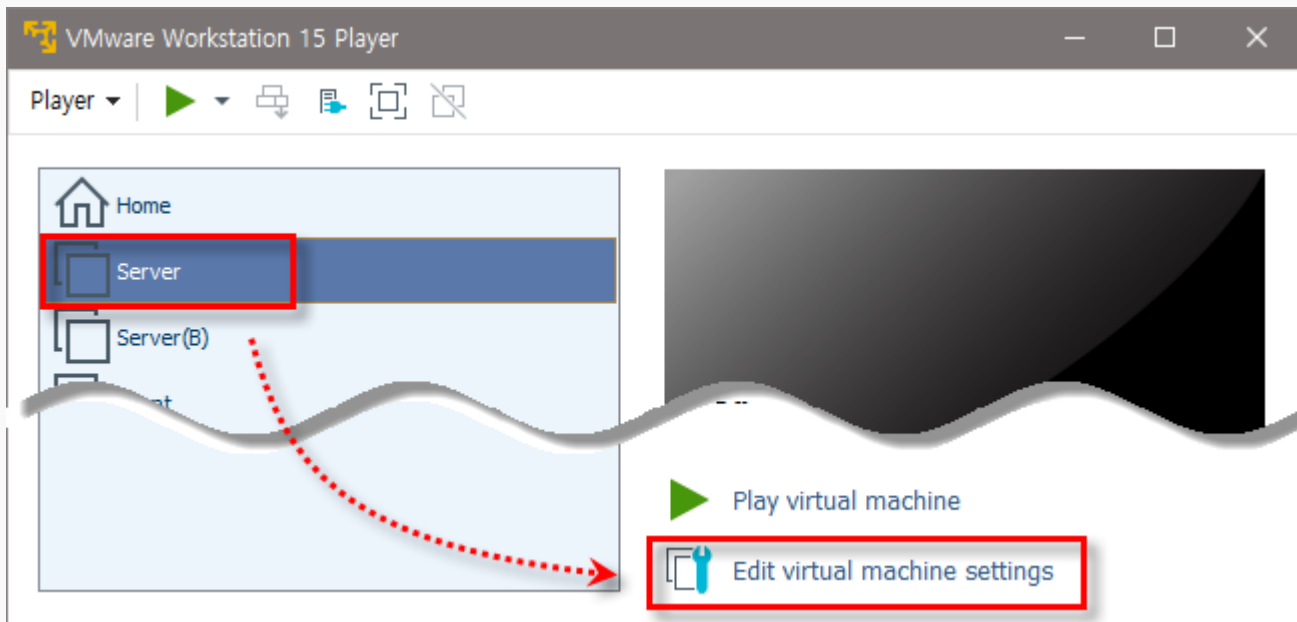




하드디스크 추가하기

1 개의 디스크 추가하기

- ① Server 가상머신 power off 상태에서 VMware 화면의 가상머신 목록에서 Server 선택, [Edit virtual machine settings] 클릭



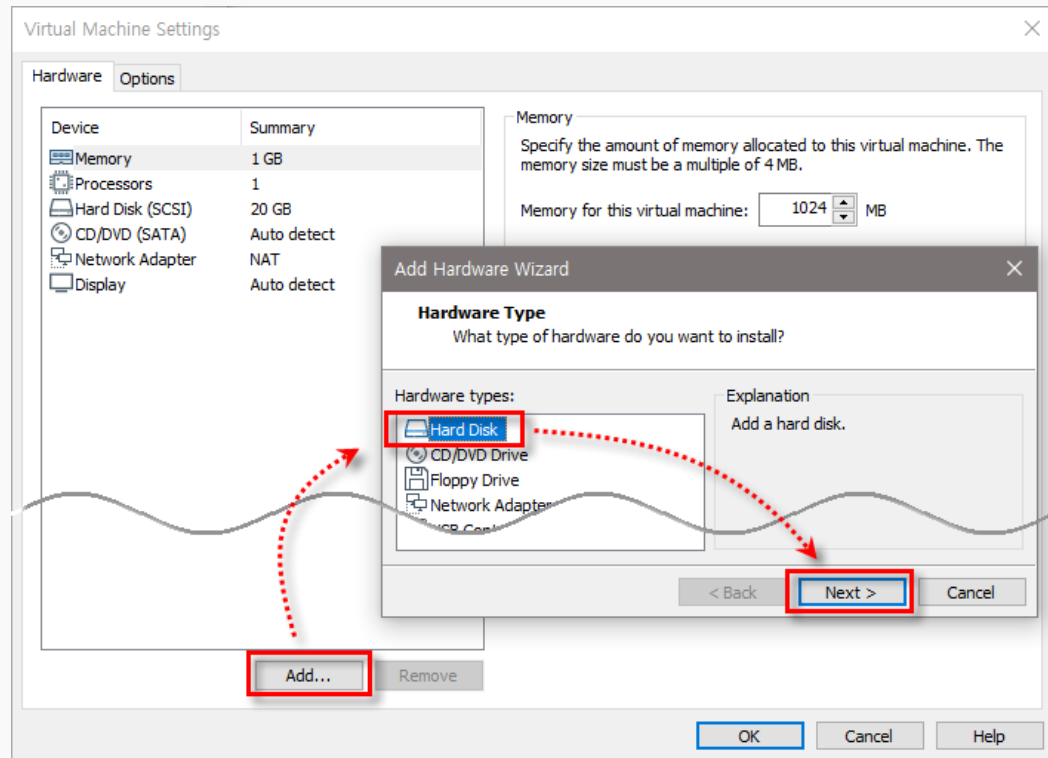
리눅스 디스크 관리



하드디스크 추가하기

1 개의 디스크 추가하기

- ② [Virtual Machine Settings] 창에서 왼쪽 아래의 <Add> 클릭
- ③ [Hardware Type] 창에서는 'Hard Disk'가 선택된 상태로 <Next> 클릭

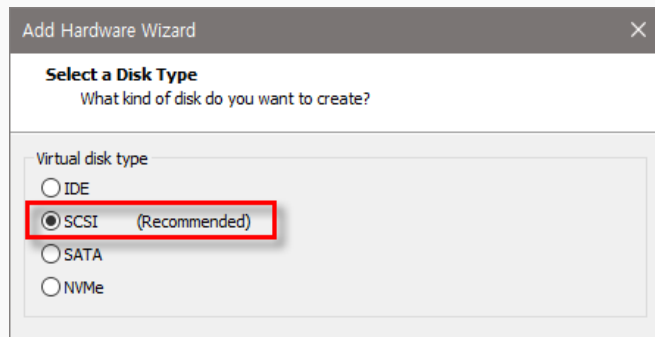




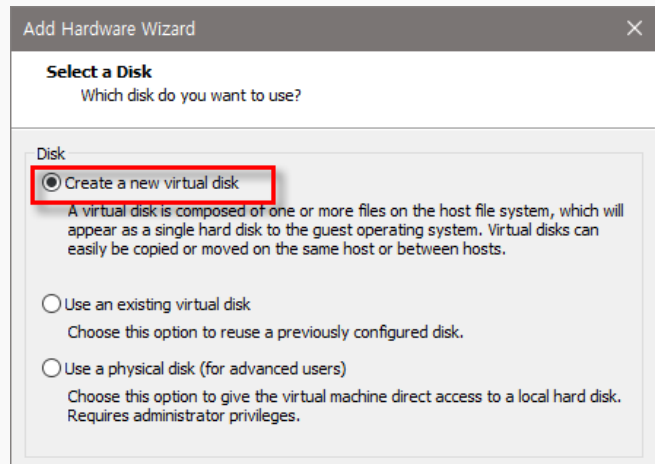
하드디스크 추가하기

1 개의 디스크 추가하기

④ Virtual disk type으로 'SCSI (Recommended)'가 선택된 상태로 <Next> 클릭



⑤ 'Create a new virtual disk'가 선택된 상태로 <Next> 클릭

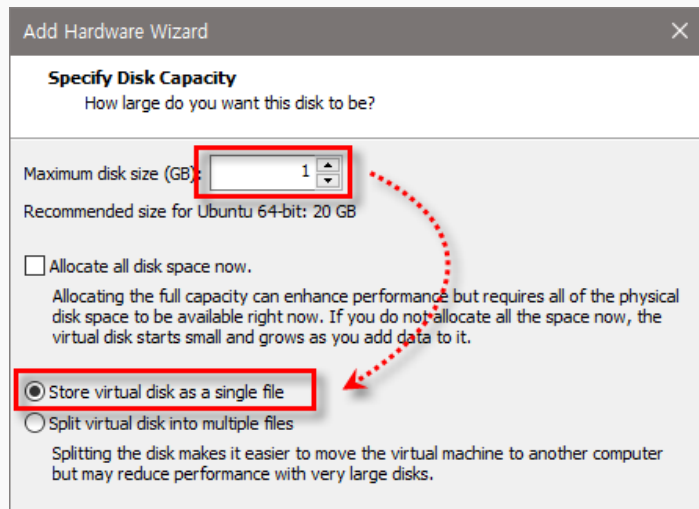




하드디스크 추가하기

1 개의 디스크 추가하기

- ⑥ Maximum disk size (GB)에 '1' 입력
'Store virtual disk as a single file' 선택 후
<Next> 클릭
- ⑦ [Specify Disk File] 창에서는 그대로 두고
<Finish> 클릭



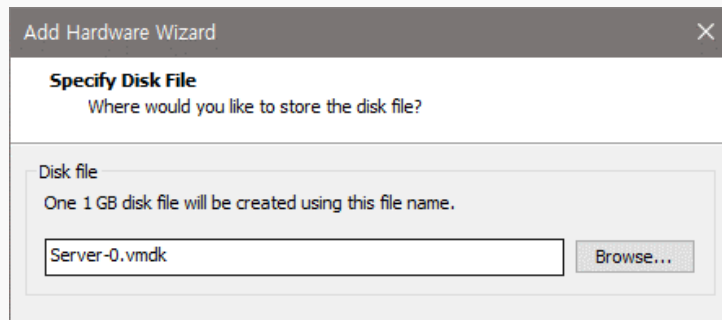
Add Hardware Wizard

Specify Disk Capacity
How large do you want this disk to be?

Maximum disk size (GB): 1
Recommended size for Ubuntu 64-bit: 20 GB

☐ Allocate all disk space now.
Allocating the full capacity can enhance performance but requires all of the physical disk space to be available right now. If you do not allocate all the space now, the virtual disk starts small and grows as you add data to it.

☒ Store virtual disk as a single file
☐ Split virtual disk into multiple files
Splitting the disk makes it easier to move the virtual machine to another computer but may reduce performance with very large disks.



Add Hardware Wizard

Specify Disk File
Where would you like to store the disk file?

Disk file
One 1 GB disk file will be created using this file name.

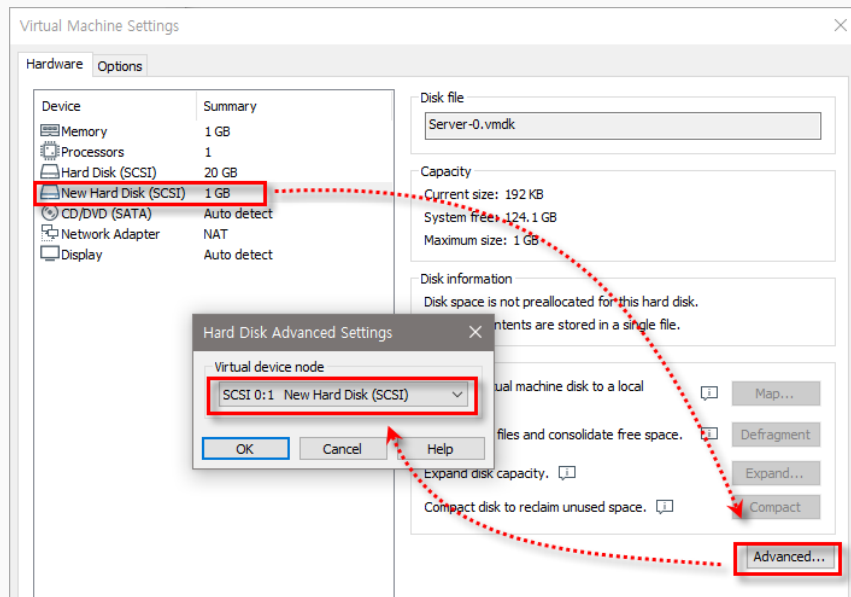
Server-0.vmdk Browse...



하드디스크 추가하기

1 개의 디스크 추가하기

- ⑧ [Virtual Machine Settings] 창의 왼쪽에 1GB 용량의 SCSI 디스크가 추가됨 새로 장착한 SCSI 디스크
선택 → <Advanced> 클릭 → <OK> 클릭
- ⑨ [Virtual Machine Settings] 창에서도 <OK>를 클릭하면 설정 내용이 적용됨





하드디스크 추가하기

1 개의 디스크 추가하기

⑩ Server를 부팅

⑪ 터미널을 열고 다음 순서대로 입력

```
# fdisk /dev/sdb      -- SCSI 0:1 디스크 선택
Command: n            -- 새로운 파티션 분할
Select: p             -- Primary 파티션 선택
Partition number: 1   -- 파티션 1번 선택(Primary 파티션은 최대 4개까지 생성 가능)
First sector:  -- 시작 섹터 번호 입력(파티션 하나만 할당하므로 첫 섹터로 설정)
Last sector:  -- 마지막 섹터 번호 입력(파티션 하나만 할당하므로 마지막 섹터로 설정)
Command: p            -- 설정 내용 확인
Command: w            -- 설정 내용 저장
```


리눅스 디스크 관리



하드디스크 추가하기

1 개의 디스크 추가하기

```
root@server: ~
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
root@server:~# fdisk /dev/sdb

Welcome to fdisk (util-linux 2.31.1).
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 0x6a85bf8d.

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

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

Command (m for help): p
Disk /dev/sdb: 1 GiB, 1073741824 bytes, 2097152 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: 0x6a85bf8d



| Device    | Boot | Start | End     | Sectors | Size  | Id | Type  |
|-----------|------|-------|---------|---------|-------|----|-------|
| /dev/sdb1 |      | 2048  | 2097151 | 2095104 | 1023M | 83 | Linux |



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

root@server:~#
```



하드디스크 추가하기

1 개의 디스크 추가하기

⑫ 포맷 : fdisk로 나눈 파티션인 /dev/sdb1을 ext4 파일 시스템으로 포맷

mkfs -t 파일시스템 파티션장치 명령 또는 mkfs.파일시스템 파티션장치 명령 입력

mkfs.ext4 /dev/sdb1 명령 실행

```
root@server: ~  
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)  
root@server:~# mkfs.ext4 /dev/sdb1  
mke2fs 1.44.1 (24-Mar-2018)  
Creating filesystem with 261888 4k blocks and 65536 inodes  
Filesystem UUID: a458eb81-2957-4bc8-a3f9-b2f9a9bfff547  
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@server:~#
```



하드디스크 추가하기

1 개의 디스크 추가하기

⑫ /mydata 디렉토리에 마운트하기

mkdir /mydata 로 디렉토리를 만들

mount -t ext4 /dev/sdb1 /mydata 명령으로 포맷이 완료된 /dev/sdb1 장치를 /mydata 디렉터리에 마운트

⑬ /mydata 디렉토리로 이동하여 ls로 확인. /boot/vmlinuz-4.18.0-15-generic 파일을 현재 디렉토리인 .으로 복사

⑭ 복사되었음을 확인

```
root@server:~# mkdir /mydata
root@server:~# mount -t ext4 /dev/sdb1 /mydata
root@server:~# cd /mydata/
root@server:/mydata# ls
lost+found
root@server:/mydata# cp /boot/vmlinuz-4.18.0-15-generic .
root@server:/mydata# ls
lost+found vmlinuz-4.18.0-15-generic
root@server:/mydata#
```



하드디스크 추가하기

1 개의 디스크 추가하기

- ⑮ 명령어 `df -h` 를 통해 디스크정보를 확인하여
/mydata의 상태 확인.

/dev/sdb1 장치가 991MB 용량에 11MB를 사용하여
2% 용량을 사용하였으며 마운트 포인트는 /mydata

```
root@server:/mydata# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            955M    0 955M   0% /dev
tmpfs           197M  1.5M  196M   1% /run
/dev/sda1       20G   6.2G   13G  33% /
tmpfs           985M    0 985M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           985M    0 985M   0% /sys/fs/cgroup
/dev/loop0      3.8M   3.8M    0 100% /snap/gnome-system-monitor/127
/dev/loop1      4.3M   4.3M    0 100% /snap/gnome-calculator/544
/dev/loop2      15M    15M    0 100% /snap/gnome-characters/399
/dev/loop3      15M    15M    0 100% /snap/gnome-logs/45
/dev/loop4      141M  141M    0 100% /snap/gnome-3-26-1604/74
/dev/loop5      161M  161M    0 100% /snap/gnome-3-28-1804/116
/dev/loop6      91M   91M    0 100% /snap/core/6350
/dev/loop7      1.0M   1.0M    0 100% /snap/gnome-logs/81
/dev/loop8      3.8M   3.8M    0 100% /snap/gnome-system-monitor/57
/dev/loop9      45M   45M    0 100% /snap/gtk-common-themes/1440
/dev/loop10     35M   35M    0 100% /snap/gtk-common-themes/818
/dev/loop11     92M   92M    0 100% /snap/core/8592
/dev/loop12     141M  141M    0 100% /snap/gnome-3-26-1604/98
/dev/loop13     2.3M   2.3M    0 100% /snap/gnome-calculator/260
/dev/loop14     55M   55M    0 100% /snap/core18/1668
/dev/loop15     13M   13M    0 100% /snap/gnome-characters/139
tmpfs           197M  20K  197M   1% /run/user/0
/dev/sda1       1.9G   1.9G    0 100% /media/root/Ubuntu 18.04.2 LTS amd64
/dev/sdb1       991M   11M  914M   2% /mydata
```



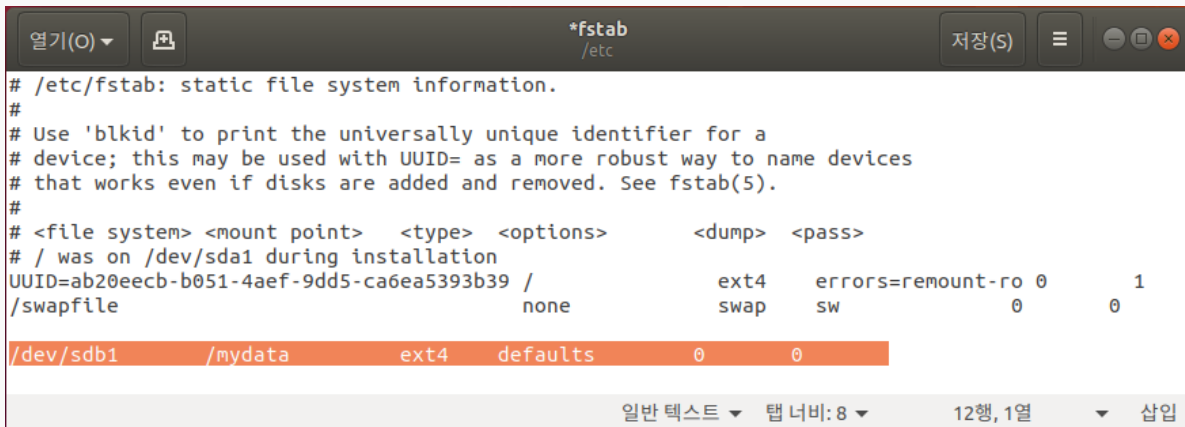
하드디스크 추가하기

1 개의 디스크 추가하기

- ⑩ 마운트를 해제할 수 있으며 명령어는 `umount /mydata` 또는 `umount /dev/sdb1`

```
root@server:/mydata# pwd
/mydata
root@server:/mydata# cd ..
root@server:/# umount /mydata/
```

- ⑪ 자동 마운트 하기. `/etc/fstab` 파일에 다음 내용을 추가하면 재부팅시 자동으로 마운트. 수정한 `/etc/fstab` 파일 저장후 `reboot` 명령으로 재부팅



```
# /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/sda1 during installation
UUID=ab20eecb-b051-4aef-9dd5-ca6ea5393b39 / ext4 errors=remount-ro 0 1
/swapfile none swap sw 0 0
/dev/sdb1 /mydata ext4 defaults 0 0
```



하드디스크 추가하기

1 개의 디스크 추가하기

- ⑮ 재부팅후 터미널에서 df -h 로 디스크 상태를 확인하면 /mydata가 자동으로 마운트되어 있음
- ⑯ /mydata를 확인하여 재부팅 전에 저장해두었던 파일 확인

```
root@server:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            955M   0 955M   0% /dev
tmpfs           197M  1.5M  196M   1% /run
/dev/sda1       20G   6.2G  13G  33% /
tmpfs           985M   0 985M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           985M   0 985M   0% /sys/fs/cgroup
/dev/loop0      35M   35M   0 100% /snap/gtk-common-themes/818
/dev/loop1      4.3M  4.3M   0 100% /snap/gnome-calculator/544
/dev/loop2      15M   15M   0 100% /snap/gnome-characters/399
/dev/loop3      15M   15M   0 100% /snap/gnome-logs/45
/dev/loop4      91M   91M   0 100% /snap/core/6350
/dev/loop5      13M   13M   0 100% /snap/gnome-characters/139
/dev/loop7      3.8M  3.8M   0 100% /snap/gnome-system-monitor/57
/dev/loop6      2.3M  2.3M   0 100% /snap/gnome-calculator/260
/dev/loop8      1.0M  1.0M   0 100% /snap/gnome-logs/81
/dev/loop9      55M   55M   0 100% /snap/core18/1668
/dev/loop10     92M   92M   0 100% /snap/core/8592
/dev/loop11     161M  161M   0 100% /snap/gnome-3-28-1804/116
/dev/loop12     3.8M  3.8M   0 100% /snap/gnome-system-monitor/127
/dev/loop13     141M  141M   0 100% /snap/gnome-3-26-1604/74
/dev/loop14     45M   45M   0 100% /snap/gtk-common-themes/1440
/dev/loop15     141M  141M   0 100% /snap/gnome-3-26-1604/98
/dev/sdb1       991M   11M  914M   2% /mydata
tmpfs           197M  20K  197M   1% /run/user/0
/dev/sr0        1.9G  1.9G   0 100% /media/root/Ubuntu 18.04.2 LTS amd64
root@server:~# ls /mydata/
lost+found vmlinuz-4.18.0-15-generic
```



하드디스크 추가하기

실전 문제

- ① 2GB크기의 하드 디스크 1개를 추가하시오.
- ② 이 디스크를 1GB크기의 파티션 2개로 나누고 첫 번째 파티션은 ext3로 포맷하고 두번째 파티션은 fat로 포맷하시오.
- ③ 첫 번째 파티션은 /root/ext3_fs 디렉토리에 마운트하고 두 번째 파티션은 /root/fat_fs 디렉토리에 마운트하시오.



디스크 관리 명령어

mount

mount

- **기능** 파일 시스템을 마운트한다.
- **형식** `mount [옵션] [장치명 또는 마운트 포인트]`
- **옵션**
 - t 파일 시스템 종류: 파일 시스템 종류를 지정한다.
 - o 마운트 옵션: 마운트 옵션을 지정한다.
 - f: 마운트할 수 있는지 점검만 한다.
 - r: 읽기만 가능하게 마운트한다(-o ro와 동일).
- **사용 예** `mount mount /dev/sdb1 / mount -t iso9660 /dev/cdrom /mnt/cdrom`

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`



디스크 관리 명령어

mount

표 7-4 다양한 장치 마운트의 예

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



디스크 관리 명령어



df

- 파일 시스템별 디스크 사용량 확인하기

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 종류도 출력한다.
- **사용 예**
 - df
 - df -h



디스크 관리 명령어



du

- 디렉터리나 사용자별 디스크 사용량 확인하기

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
```



디스크 관리 명령어



du

- 현재 디렉토리의 디스크 사용량을 출력

```
user1@myubuntu:~$ pwd
/home/user1
user1@myubuntu:~$ du
4      ./다운로드
284    ./cache/gstreamer-1.0
20     ./cache/update-manager-core
4      ./cache/evolution/tasks/trash
8      ./cache/evolution/tasks
4      ./cache/evolution/calendar/trash
8      ./cache/evolution/calendar
(생략)
8      ./linux_ex/ch6
124    ./linux_ex
36004  .
user1@myubuntu:~$
```



디스크 관리 명령어



du

- s 옵션 : 전체 디스크 사용량 출력하기

```
user1@myubuntu:~$ du -s
36004  .
```

```
user1@myubuntu:~$ sudo du -s /etc
12956  /etc
user1@myubuntu:~$
```

- 특정 사용자의 디스크 사용량 출력하기

```
user1@myubuntu:~$ du -sh ~user1
36M   /home/user1
user1@myubuntu:~$
```



디스크 관리 명령어

fsck

- 파일 시스템 검사하고 복구하기

- ✓ 파일 시스템은 부적절한 시스템 종료나 전원의 불안정, 소프트웨어 오류, 하드웨어 오작동 등 다양한 이유로 손상될 수 있음
- ✓ 손상된 파일 시스템의 용량을 확인할 뿐만 아니라 파일 시스템의 상태를 점검하고 문제가 있을 때 복구해야 함

- fsck 명령으로 파일 시스템 검사하기

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 대답하게 한다.
 - a: 파일 시스템 검사에서 문제를 발견했을 때 자동으로 복구한다.
- **사용 예**

```
fsck /dev/sdb1  
fsck -f /dev/sdb1
```



디스크 관리 명령어



fsck

- 일반적인 파일 시스템 검사

```
user1@myubuntu:~$ sudo fsck /dev/sdd1
fsck from util-linux 2.30.1
e2fsck 1.43.5 (04-Aug-2017)
/dev/sdd1: clean, 11/76912 files, 11777/307200 blocks
user1@myubuntu:~$
```

- 파일 시스템 강제 검사

```
user1@myubuntu:~$ sudo fsck -f /dev/sdd1
fsck from util-linux 2.30.1
e2fsck 1.43.5 (04-Aug-2017)
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: 11/76912 files (0.0% non-contiguous), 11777/307200 blocks
user1@myubuntu:~$
```



디스크 관리 명령어



dd

- 입력파일의 지정한 크기만큼의 블록을 출력 파일에 복사
- 이 명령은 임베디드 리눅스에서 루트 파일 시스템을 제작할 때 자주 사용됨.

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 수이다.
 - `count=블록 수`: 블록 수만큼만 복사한다
- **사용 예** `dd if=/dev/zero of=/dev/sdd1 bs=4096 count=20`



디스크 관리 명령어



dd

- /var/log/dmesg 파일의 내용을 읽고 /tmp/test파일에 쓰는데 1024바이트 크기로 10번 반복

```
root@server:~# ls -alF /var/log/kern.log
-rw-r----- 1 syslog adm 1719916  2월 20 13:33 /var/log/kern.log
root@server:~# dd if=/var/log/kern.log of=/tmp/test bs=1024 count=10
10+0 레코드 들어옴
10+0 레코드 나감
10240 bytes (10 kB, 10 KiB) copied, 0.00029033 s, 35.3 MB/s
root@server:~# ls -alF /tmp/test
-rw-r--r-- 1 root root 10240  2월 20 14:08 /tmp/test
root@server:~# █
```

Q & A

Thank You