SILCROAD SOFT

일일 업무 사항 정리

작성자	제품팀 이민성 인턴
업무 일시	2022.08.18

세부 사항

1. 업무 내역 요약 정리

목표 내역	Done & Plan	To-do
1. 리눅스 환경에 대해 이해 2. 리눅스 명령어 숙달 3. 계정 및 디렉토리 관리 공부 4.오라클 데이터베이스 설치과정 복기	오늘은 그동안 배운 것을 다시 한번 공부하고 실행해보며 상기시켰습니다. 추가로 계정관리, 그룹관리, grep 명령어, 리눅스 종료 명령어, GID, UID, 쉘, 디렉토리, 패키지에 대해 공부하였습니다. 그리고 지금까지 정리한 것을 깔끔하게 다시 한번 정리하고 추가로 공부한 것들을 포함해 저만의 리눅스 사전을 만들었습니다. 리눅스를 공부하면서 앞으로 리눅스를 많이 사용할 텐데 그때그때 인터넷을 찾기보다는 제가 만든 자료를 보며 찾으면 시간도 절약하고 빠르게 습득할 수 있을 것 같습니다. 전에 작성했던 제가 공부한 자료들과 거의 같은 내용이지만 복사 붙여넣기를 하지 않고 손수 하나하나 다시 쓰며 상기시켰습니다. 앞으로 공부를 하며 부족한 내용을 점차 채워가면서 리눅스 명령어를 습득하면 좋을 것 같습니다. 앞으로 리눅스를 활용을 할 수 있는 예제를 찾아서 활용하고 오라클 데이터베이스를 설치했던 과정을 복기하면서 추가로 모르는 내용들을 공부할 예정입니다.	- 리눅스 환경에 대한 이해 -- 계정 및 디렉토리 관리 -- 기본 명령어 - 설치 -- 과정 복기

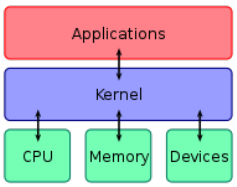
2. 내용 세부 (업무 세부 내역 정리 및 기타 사항 정리)

1. 유닉스 (unix)

- 유닉스 : 주로 서버용 컴퓨터에서 사용되는 운영체제이다.
- 유닉스의 특징
  1. 시분할 시스템*(Time Sharing System)을 위해 설계된 대화식 운영체제이다.
     *시분할 시스템 : 각 사용자들에게 컴퓨터 자원을 시간적으로 분할하여 사용할 수 있게 해주는 시스템
  2. 대부분 C언어로 작성되어 있어 이식성이 높으며 장치, 프로세스 간의 호환성이 높다.
  3. 크기가 작고 이해하기가 쉽다.
  4. 다중 사용자, 다중 작업을 지원한다.
  5. 많은 네트워킹 기능을 제공하므로 통신망 관리용 운영체제로 적합하다.
  6. 트리구조의 파일 시스템을 가지고 있다.
  7. 전문적인 프로그램 개발에 용이하다.
  8. 다양한 유틸리티 프로그램들이 존재한다.
- 유닉스와 리눅스의 차이
  1. 리눅스는 무료이나 유닉스는 아니다.
  2. 리눅스는 오픈소스이기에 개발자가 프로젝트에 기여할 수 있으며 버그수정, 보안 패치 및 보다 강력한 시스템을 제공할 수 있다. 유닉스 시스템은 솔라리스와 맥 OS와 같이 여전히 진화하고 있지만, 리눅스는 더 큰 사용자 기반을 가지고 있다.

2. 리눅스

- 리눅스 : 유닉스에서 파생되었으며 컴퓨터 운영체제의 한 종류이자, 커널* 자체를 의미하기도 한다.
- 커널
  - 정의 : 컴퓨터 운영체제의 핵심이 되는 컴퓨터 프로그램으로, 시스템의 모든 것을 완전히 제공한다. 운영 체제의 다른 부분 및 응용 프로그램 수행에 필요한 여러가지 서비스를 제공한다. 핵심이라고도 한다.



  - 커널의 역할 : 커널은 운영체제의 핵심 부분이므로, 커널의 역할 역시 운영체제의 핵심 역할이라 할 수 있다.
    1. 보안 : 커널은 컴퓨터 하드웨어와 프로세스의 보안을 책임진다.
    2. 자원 관리 : 한정된 시스템 자원을 효율적으로 관리하여 프로그램의 실행을 원활하게 한다.
    3. 추상화 : 같은 종류의 부품에 대해 다양한 하드웨어를 설계할 수 있기 때문에 하드웨어에 직접 접근하는 것은 문제를 매우 복잡하게 만들 수 있다. 일반적으로 커널은 운영체제의 복잡한 내부를 감추고 깔끔하고 일관성 있는 인터페이스를 하드웨어에 제공하기 위해 몇 가지 하드웨어 추상화(같은 종류의 장비에 대한 공통 명령어의 집합)들로 구현된다. 이 하드웨어의 추상화는 프로그래머가 여러 장비에서 작동하는 프로그램을 개발하는 것을 돕는다.

- 리눅스의 장단점
  1. 소스 코드가 공개 되어있는 자유 소프트웨어와 오픈 소스 개발의 가장 유명한 표본이다. 때문에 윈도우는 MS사의 라이선스를 구매해야 사용이 가능한 반면 리눅스는 별도의 비용이 발생하지 않는다.
  2. 윈도우의 경우 운영체제 내 소스코드의 저작권이 MS에 있어 일반적으로 소스 수정이 불가능하다. 이로 인해 심각한 취약점이 발생하였을 때 MS사에서 패치가 나오지 않거나 중단되었을 경우에는 안전을 보장받을 수 없다. 반면 리눅스는 소스가 공개 되어있기 때문에 취약점 노출 시 비교적 빠른 보안 업데이트가 진행될 수 있다. 하지만 공개 소프트웨어의 특성상 소스를 분석하여 취약점을 노출시킬 수 있는 악의적인 사용자가 존재할 가능성이 높다.
  3. 다중 사용자, 다중 작업, 다중 스레드를 지원하는 네트워크 운영체제로 여러 사람이 하나의 리눅스 시스템에 접속하며 다수의 프로그램을 동시에 실행할 수 있다.
  4. 커널을 포함하여 리눅스 자체가 어셈블리와 C로 개발되어 있어 이후에 만들어진 C++, Java. Perl, PHP, Python, Ruby, Lua, Go 언어 등을 모두 호환한다.
  5. 지원하는 시스템 장치의 호환성이 좋지 않고, 인터페이스 환경에서 원도우에 비해 불편하다.
- CLI, TUI, GUI 환경
  - CLI : Command Line Interface로 명령어를 줄로 입력하여 소통한다 라는 뜻이다.
    1. 운영체제 안에 있는 쉘이 가지고 있는 특정 명령어를 통해 운영체제를 컨트롤한다.
    2. 윈도우의 cmd나 리눅스의 터미널이 대표적이다.
    3. 키보드와 명령어를 사용할 수 있다.
  - TUI :Text User Interface로 글로 사용자가 소통한다 라는 뜻이다.
    1. 리눅스 안의 vi(vim) 에디터가 대표적이다.
    2. CLI와 비슷하지만 다르다. 명령어를 사용해도 전혀 안 되며, 메모장과 비슷하다. 메모장과 차이점은 키보드로만 컨트롤 할 수 있는 것이다.
  - GUI : Graphic User Interface로 그래픽으로 사용자가 소통한다 라는 뜻이다.
    1. 키보드, 마우스 모두 사용할 수 있다.
    2. 한 눈에 보이고 제일 편하다.

       *GUI가 제일 편한데 왜 리눅스에서는 잘 사용하지 않을까?

       보통 리눅스는 서버용으로 CLI환경을 많이 사용한다. 왜냐하면 그래픽으로 사용하면 자원을 많이 잡아먹어 부하가 많이 걸리기 때문이다.
- 리눅스를 서버로 사용하는 이유
  1. 무료 오픈소스
  2. 우수한 보안성
  3. 구조 안전성 : 패치 이후에 OS를 재부팅 할 필요가 없다. 기업에서 사용하는 웹 서버는 주로 하루 24 시간 365 일 내내 가동 중이어야 하는 경우가 많은데, OS가 재부팅 된다면 웹서버를 다시 가동하고 서버 안정화를 시키는데 꽤 많은 시간을 들여야만 한다.
  4. 관리자의 관리가 용이하다.
- 리눅스 배포판 (리눅스와 CentOs의 관계)

  리눅스 배포판은 리눅스 커널, GNU 소프트웨어 및 여러가지 자유 소프트웨어로 구성된 운영체제이다. 유닉스 계열 OS들과는 달리 유닉스에 기반을 두지 않고 기술적으로 독립적인 환경에서 유닉스를 모방하여 개발되었다. 회사 차원에서 관리하고 배포하는 레드헷 리눅스, 우분투, 수세 리눅스 등도 있고, 커뮤니티 차원에서 관리하고 배포하는 데비안, 젠투 리눅스, 페도라 등이 있다. 여러 소프트웨어를 모으고 시험하여 배포판을 만든다.

- Vi (Vim) 에디터
  - Vi (Visual editor) : 유닉스 환경에서 가장 많이 쓰이는 문서 편집기
  - Vi의 명령어 :
    1. i : 입력모드
    2. esc : 입력상태에서 esc키를 누르면 명령모드로 바뀌게 된다.
    3. G : 맨 아래로
    4. o : 아래 한 줄 추가 후 입력
    5. 명령모드 진입 후 q! : 강제종료
    6. 명령모드 진입 후 wq! : 저장 후 종료

3. 리눅스의 기본 명령어

1. pwd : 현재 작업중인 디렉토리의 위치를 출력

```
[oracle@lms test]$ pwd
/app/oracle/test
```

2. ls : 현재 위치의 파일 목록 조회

```
[oracle@lms test]$ ls -ltr
합 계  20
-rw-r--r-- 1 oracle oinstall   73  8월  17 10:57 HelloWorld.c
-rwxr-xr-x 1 oracle oinstall 8360  8월  17 10:58 HelloWorld
-rw-r--r-- 1 oracle oinstall   23  8월  17 13:24 HelloWorld.py
-rw-r----x 1 oracle oinstall    0  8월  17 17:20 test1
```

-l	파일의 상세정보
-a	숨김 파일 표시
-t	파일들을 생성시간 순 (제일 최신 것부터)으로 표시
-tr	파일들을 생성시간 순 (제일 오래된 것부터)으로 표시
-f	파일 표시 시 마지막 유형에 나타내는 파일명을 끝에 표시 (/ : 디렉터리, * : 실행파일, @ : 링크 등)

3. cd : 디렉터리 이동

```
[oracle@lms test]$ cd ..
[oracle@lms oracle]$
```

디렉터리 경로	이동하려는 디렉터리로 이동
~	홈 디렉터리로 이동
/	최상위 디렉터리로 이동
.	현재 디렉터리
..	상위 디렉터리로 이동
-	이전 경로로 이동

4. touch : 0 바이트 파일 생성, 파일의 날짜와 시간을 수정

```
[oracle@lms test]$ touch file
[oracle@lms test]$ ls -lt
합 계  20
-rw-r--r-- 1 oracle oinstall    0  8월  18 11:11 file
```

filename	filename의 파일을 생성
-c filename	filename의 시간을 현재 시간으로 갱신
-t 202110292680 filename	filename의 시간을 날짜 정도로 갱신 (202110291608 → 2021.10.29.16:08)
-r oldfile newfile	newfile의 날짜 정보를 oldfile의 날짜 정보와 동일하게 변경

5. mkdir : 디렉터리 생성

```
[oracle@lms oracle]$ mkdir dirname
[oracle@lms oracle]$ ls -lt
합 계  2487200
drwxr-xr-x  2 oracle oinstall          6  8월  18 11:18 dirname
```

dirname	dirname이라는 디렉터리 생성
dir1 dir2	한 번에 여러 개의 디렉터리 생성
-p dirname/sub_dirname	dirname이라는 디렉터리 생성, sub_dirname이라는 하위 디렉터리도 생성
-m 700 dirname	특정 권한을 갖는 디렉터리 생성

6. cp : 파일복사

```
[oracle@lms test]$ cp file file2
[oracle@lms test]$ ls -lt
합 계  20
-rw-r--r-- 1 oracle oinstall    0  8월  18 11:21 file2
-rw-r--r-- 1 oracle oinstall    0  8월  18 11:20 file
```

<file을 file2 라는 이름으로 복사>

file1 file2	file1 을 file2 라는 이름으로 복사 (file2 가 없을 경우 file2 를 생성)
-f file1 file2	강제 복사 (file2 라는 파일이 이미 있을 경우 강제로 기존 file2 를 지우고 복사 진행)
-r dir1 dir2	디렉터리 복사 (폴더 안의 모든 하위 경로와 파일들을 복사)

7. mv : 파일 이동

```
[oracle@lms test]$ ls -t
file  test1  HelloWorld.py  HelloWorld  HelloWorld.c
[oracle@lms test]$ mv file /app
[oracle@lms test]$ cd ..
[oracle@lms oracle]$ cd ..
[oracle@lms app]$ ls -t
file  oracle  oraInventory
```

file1 file2	file1 파일을 file2 파일로 변경
file1 /dir	file1 파일을 dir 디렉터리로 이동
file1 file2 /dir	여러 개의 파일을 dir 디렉터리로 이동
dir1/ dir2/	dir1 디렉터리를 dir2 디렉터리로 변경

메모 포함[이1]: 1 차 변경사항

8. rm : 파일삭제

```
[oracle@lms test]$ ls
HelloWorld  HelloWorld.c  HelloWorld.py  test1
[oracle@lms test]$ rm test1
[oracle@lms test]$ ls
HelloWorld  HelloWorld.c  HelloWorld.py
```

file1	file1 을 삭제
-f file1	file1 을 강제 삭제
-r dir1	dir1 디렉터리 삭제 (디렉터리는 -r 옵션 없이 삭제 불가)
rmdir dir1	dir1 디렉터리 삭제

9. cat : 파일의 내용을 화면에 출력, 리다이렉션 기호(>)를 사용하여 새로운 파일 생성

```
[oracle@lms test]$ cat HelloWorld.py
print("Hello, World!")
```

file1	file1 의 내용을 출력
file1 file2	file1 과 file2 의 내용을 출력
file1 file2 | more	file1 과 file2 의 내용을 페이지 별로 출력
file1 file2 | head	file1 과 file2 의 내용을 처음부터 10 번째 줄까지만 출력
file1 file2 | tail	file1 과 file2 의 내용을 끝에서부터 10 번째 줄까지만 출력

- redirection : 화면의 출력 결과를 파일로 저장

>	기존에 있는 파일 내용을 지우고 저장
>>	기존 파일 내용 뒤에 덧붙여서 저장
<	파일의 데이터를 명령에 입력

cat file file2 > file3 : file1, file2 의 명령 결과를 합쳐서 file3 에 저장

cat file4 >> file3 : file3 에 file4 의 내용 추가

cat < file1 > file2 : file1 의 출력 결과를 file2 에 저장

10. alias : 자주 사용하는 명령어들을 별명으로 정의하여 쉽게 사용할 수 있도록 설정

alias 별명	명령어 정의 (alias ls='ls -ltr' : ls를 실행하면 ls -ltr이 실행)
unalias 별명	별명이라는 alias를 해제

```
[oracle@lms test]$ alias ls='ls -ltr'
[oracle@lms test]$ ls
합 계  20
-rw-r--r-- 1 oracle oinstall   73  8월  17 10:57 HelloWorld.c
-rwxr-xr-x 1 oracle oinstall 8360  8월  17 10:58 HelloWorld
-rw-r--r-- 1 oracle oinstall   23  8월  17 13:24 HelloWorld.py
[oracle@lms test]$ unalias ls
[oracle@lms test]$ ls
HelloWorld  HelloWorld.c  HelloWorld.py
```

11. grep : 리눅스 문자열 검색하기

```
[oracle@lms test]$ ls -ltr | grep py
-rw-r--r-- 1 oracle oinstall   23  8월  17 13:24 HelloWorld.py
```

12. shutdown : 리눅스 종료 명령어

```
[root@lms ~]# shutdown -r now
Connection closing...Socket close.

Connection closed by foreign host.

Disconnected from remote host(lms) at 11:50:32.
```

-h now	지금 즉시 종료
-r now	지금 즉시 재부팅

- 새로운 파일을 만드는 방법
  1. vi newfile : vi 편집기 상태로 들어감
  2. touch newfile : 빈 파일만 생성됨
  3. cat > newfile : vi 편집기 상태로 들어감, 문서 작성후 ctrl + d로 빠져나옴

- 파일 내용만 보기
  1. cat filename : 파일의 내용을 모두 보여줌
  2. head -n filename : n줄 만큼 위에서부터 보여줌
  3. tail -n filename : n줄 만큼 아래에서부터 보여줌

4. 리눅스에서 C 작성하기

1. GCC* 설치
   yum install gcc
2. C파일 생성
   vi HelloWorld.c
3. C코드 작성

```c
#include <stdio.h>
int main() {
        printf("Hello, World!\n");
        return 0;
}
```

4. 컴파일 하기
   gcc HelloWorld.c -o HelloWorld (o옵션으로 실행파일명을 지정해줄 수 있음)
   gcc HelloWorld.c (a.out이라는 실행파일이 생성됨)
5. ./HelloWorld로 실행

```
[oracle@lms test]$ gcc --version
gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-44)
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

[oracle@lms test]$ vi HelloWorld.c
[oracle@lms test]$ gcc HelloWorld.c -o HelloWorld
[oracle@lms test]$ ./HelloWorld
Hello, World!
```

- GCC
  - GCC : GNU 컴파일러* 모음 (GNU Compiler Collection)의 약자고. GNU 프로젝트의 일환으로 개발되어 널리 쓰이고 있는 컴파일러이다. 공식적으로 지원하는 언어는 C(gcc), C++(g++), Objective-C(gobjc), Fortran(gfortran), Ada(gnat), Go(gccgo), D(gdc)이다. Java(gcj)는 GCC 7.1 버전부터 지원이 중단되었다. GNU 진영에서는 GCC로 컴파일을 하고 Make를 이용해 빌드하는 것이 일반적이다.
  - 컴파일러(compiler)는 특정 프로그래밍 언어로 쓰여 있는 문서를 다른 프로그래밍 언어로 옮기는 프로그램을 말한다. 원래의 문서를 소스 코드 혹은 원시 코드라고 부르고, 출력된 문서를 목적 코드라고 부른다. 목적 코드는 주로 다른 프로그램이나 하드웨어가 처리하기에 용이한 형태로 출력되지만 사람이 읽을 수 있는 문서 파일이나 그림 파일 등으로 옮기는 경우도 있다. 원시 코드에서 목적 코드로 옮기는 과정을 컴파일(compile)이라고 한다.
  - 옵션

gcc -v	현재 사용하는 gcc의 버전을 나타냄
gcc -o	실행파일의 이름을 지정하는 옵션 (gcc test.c -o test : test.c는 소스파일이며 test이라는 이름으로 실행파일을 생성하는 것임)

- yum과 sudo
  - yum : Yellowdog Updater Modified의 약자로, RPM기반의 시스템을 위한 자동 업데이터이자 소프트웨어와 같은 패키지* 설치 / 삭제 도구
    *패키지 : 리눅스 시스템에서 소프트웨어를 실행하는데 필요한 파일들이 담겨 있는 설치 파일 묶음
  - sudo : 유닉스 및 유닉스 계열 운영 체제에서 다른 사용자의 보안 권한과 관련된 프로그램을 구동할 수 있게 해주는 프로그램

5. 리눅스에서 파이썬 코드 작성하기

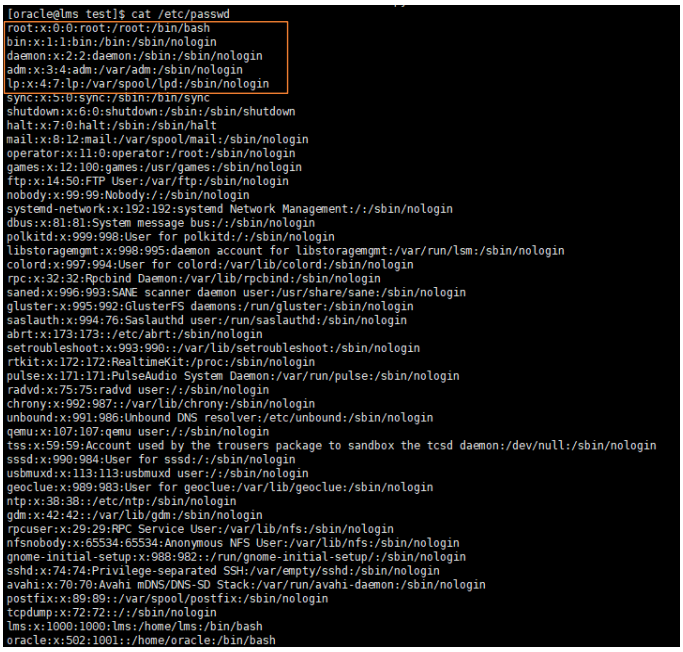
1. 파이썬 파일 생성
   vi HelloWorld.py
2. 파이썬 코드 입력
   ```
   print("Hello, World!")
   ```
3. 파이썬 파일 실행
   python HelloWorld.py

* 만약에 권한 거부(permission denied)가 출력된다면 "chmod 755 HelloWorld.py"로 HelloWorld.py 파일에 대한 권한 설정을 해야한다.

```
[oracle@lms test]$ vi HelloWorld.py
[oracle@lms test]$ python test.py
Hello, World!
```

6. 리눅스 계정 관리

- 사용자 계정 확인
  사용자 계정 관리의 핵심 파일은 /etc/passwd 파일이다. /etc/passwd 파일은 일반 편집기 파일로 열어 볼 수도 있고 편집도 가능하다. 해당 파일을 vi 편집기로 보면 보통 아래와 같은 내용들이다.



첫번째 줄만 내용을 추려서 "oracle:x:502:1001::/home/oracle:/bin/bash"를 보면

1 :2: 3 : 4 :5:/ 6 :/ 7

이렇게 7 개의 필드로 구분된다. 각각의 필드에 대한 설명은 아래와 같다.

1. 사용자 계정 ID
2. 패스워드
3. 사용자 UID*
4. 그룹 GID*
5. 계정 정보 (보통은 사용자 이름)
6. 홈 디렉토리
7. 쉘 환경

위의 정보는 사용자 계정 등록 시 등록할 수 있지만 별도로 저장하거나 /etc/passwd를 수정하여 비밀번호를 제외한 나머지를 수정하여 설정할 수도 있다. 두 번째 필드는 비밀번호 필드인데 보안유지를 위해 x로 되어있다. 비밀번호를 관리하는 파일은 /etc/shadow 파일이다.

```
[oracle@lms test]$ cat /etc/passwd | grep oracle
oracle:x:502:1001::/home/oracle:/bin/bash
```

위와 같이 "| grep oracle"을 통해 oracle이 들어간 계정을 확인할 수도 있다.

*GID : 그룹 ID

*UID : User ID (0~32767, 0 은 root유저를 나타냄)

- 사용자 계정 등록 : useradd

-c	계정설명, 대부분 사용자명 입력, (finger 명령어로 확인 가능한 간단한 사용자 설명)
-d	사용자 홈 디렉터리 경로 설정
-m	사용자 홈 디렉터리 생성
-e	사용자 계정의 사용 종료 일자
-f	사용자 계정의 유효기간 (-f 180 : 계정 생성일로부터 180 일 동안만 사용가능)
-g	사용자 계정의 로그인 그룹
-G	사용자 계정의 추가 등록 계정의 그룹명
-p	사용자 계정의 패스워드 (useradd -p 'openssl passwd abcd1234' user-1, 계정 비밀번호의 경우 암호화되어 저장되기 때문에 예시처럼 openssl passwd를 사용해 등록하지 않으면 로그인 할 수 없게된다.)
-s	사용자 계정의 로그인 쉘* 설정
-u	사용자 계정의 UID 설정

*쉘 : 명령어를 해석하는 프로그램 (커널과 사용자 간의 다리역할, 사용자로부터 명령을 받아 그것을 해석하고 프로그램을 실행하는 역할, 터미널이라고 할 수 있음)

*로그인 쉘 : 사용자가 로그인 했을 때 해당 사용자에게 적용되는 쉘

```
[root@lms ~]# useradd -d /home/user -u 600 -s /bin/csh user
[root@lms ~]# useradd -d /home/user1 -c 'test user1' -u 700 user1
```

1. 사용자 계정 추가 명령어 -d 디렉토리 -u UID -s 로그인 쉘 사용자 계정 ID
2. 사용자 계정 추가 명령어 -d 디렉토리 -c '계정 설명' -u UID 사용자 계정 ID

```
[root@lms ~]# cat /etc/passwd | grep /home/user
user:x:600:1004::/home/user:/bin/csh
user1:x:700:1005:test user1:/home/user1:/bin/bash
```

useradd를 통해서 user, user1 계정을 만들고 cat /etc/passwd를 통해서 잘 만들어졌는지 확인도 해보았다.

이러한 방법으로 사용자 계정을 생성할 수 있다. 하지만 나의 경우에 아래와 같은 에러가 발생하였다. 그 이유는 루트 권한으로 실행하지 않고 오라클 계정으로 실행을 하였기 때문이다.

```
[oracle@lms /]$ useradd -d /home/user -u 600 -s /bin/csh user
useradd: Permission denied.
useradd: /etc/passwd을 (를) 잠글 수 없습니다. 나중에 다시 시도하십시오.
```

- 사용자 계정 삭제 : userdel

```
[root@lms ~]# userdel -r user
[root@lms ~]# userdel -r user1
[root@lms ~]# cat /etc/passwd | grep /home/user
```

계정명	계정만 삭제
-r 계정명	계정과 홈 디렉터리 삭제

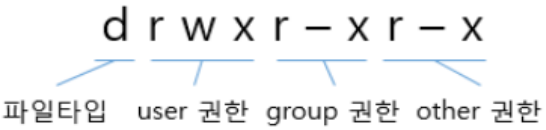
```
[oracle@lms /]$ userdel -r user1
userdel: Permission denied.
userdel: /etc/passwd을 (를) 잠글 수 없습니다. 나중에 다시 시도하십시오.
```

userdel도 마찬가지로 oracle 계정에서 실행해봤지만 에러가 발생하였다.

- 사용자 권한 변경
  - 기본적인 사용자 권한 개념

    리눅스에서 파일의 권한은 크게 3 가지로 분류된다. 파일의 소유권자, 소속 그룹, 다른 모든 사용자이다. 리눅스에서 ls -al 명령으로 파일의 권한을 확인해 볼 수 있다.

```
[oracle@lms oracle]$ ls -al
합계 2487200
drwxr-xr-x 11 oracle oinstall        245  8월  17 09:30 .
drwxr-xr-x  4 oracle oinstall         40  8월  15 19:23 ..
drwxr-x---  3 oracle oinstall         18  8월  15 19:54 admin
drwxr-xr-x  6 oracle oinstall         60  8월  15 19:55 cfgtoollogs
drwxr-xr-x  2 oracle oinstall          6  8월  16 09:51 checkpoints
drwxr-xr-x  7 oracle oinstall        136  8월  27  2013 database
drwxrwxr-x 11 oracle oinstall        128  8월  15 19:47 diag
drwxr-x---  4 oracle oinstall         30  8월  15 19:55 fast_recovery_area
drwxr-x---  3 oracle oinstall         18  8월  15 19:54 oradata
-rwxr-xr-x  1 oracle oinstall 1395582860  8월  14 23:04 p13390677_112040_Linux-x86-64_1of7.zip
-rwxr-xr-x  1 oracle oinstall 1151304589  8월  14 23:03 p13390677_112040_Linux-x86-64_2of7.zip
drwxr-xr-x  3 oracle oinstall         22  8월  14 22:59 product
drwxr-xr-x  2 oracle oinstall         65  8월  17 14:08 test
```



가장 앞자리는 파일이 디렉토리인지 또는 일반 파일인지 등 파일 타입을 나타내고 그 다음은 3 자리씩 묶어서 파일의 권한을 설정한다. 파일의 타입과 각 사용자에 따른 권한에 대한 표시의 의미는 아래의 표와 같다. 주로 보게 되는 타입은 -와 d이다.

-	plain file, 일반적인 파일이 여기에 해당한다. (실행 파일 포함)
d	directory
l	link, 다른 파일을 가리키는 링크파일
p	pipe, 두 개의 프로그램을 연결하는 파일
b	block device, 블록 단위로 하드웨어와 반응하는 파일
c	character device, 스트림 단위로 하드웨어와 반응하는 파일

파일에 대한 권한은 파일의 소유자나 그룹, 모든 사용자가 동일한 규칙을 적용 받는다. 그 의미는 아래의 표와 같다.

r(4)	읽기 (read)
w(2)	쓰기 (write)
x(1)	실행 (execute)

r/w/x가 들어갈 자리에 -가 있다면 해당 권한이 없다는 뜻이다. 예를 들어, r-x라면 읽기과 실행이 가능한 파일이라는 뜻이다.

- 사용자 권한 변경 명령어 : chmod

<방법 1>

```
[oracle@lms test]$ ls -ltr | grep test1
-rw-r--r-- 1 oracle oinstall    0  8월  17 17:20 test1
[oracle@lms test]$ chmod g+w test1
[oracle@lms test]$ ls -ltr | grep test1
-rw-rw-r-- 1 oracle oinstall    0  8월  17 17:20 test1
```

```
[oracle@lms test]$ chmod o-r test1
[oracle@lms test]$ ls -ltr | grep test1
-rw-rw---- 1 oracle oinstall    0  8월  17 17:20 test1
```

+/- 기호 중심으로 앞에는 user/group/other을 설정하고 뒤에는 읽기/쓰기/실행의 권한을 설정한다.
+는 권한을 주는 것이고, -는 권한을 삭제하는 것이다.
u : user, g : group, o : other, a : all이다. (위의 권한 설명 사진을 참조하면 쉽게 이해할 수 있다.)

<방법 2>

```
[oracle@lms test]$ ls -ltr | grep test1
-rw-rw---- 1 oracle oinstall    0  8월  17 17:20 test1
[oracle@lms test]$ chmod 777 test1
[oracle@lms test]$ ls -ltr | grep test1
-rwxrwxrwx 1 oracle oinstall    0  8월  17 17:20 test1
[oracle@lms test]$ chmod 641 test1
[oracle@lms test]$ ls -ltr | grep test1
-rw-r----x 1 oracle oinstall    0  8월  17 17:20 test1
```

방법2는 위의 권한 표 중 괄호안의 속성과 관련이 있다. 총 3자리 숫자는 각각 user, group, other과 관련이 있다. r, w, x를 각각 숫자로 바꿔보면 4, 2, 1 이 된다. 이것을 합하면 7 이며 즉 읽고 쓰고 실행하는 모든 권한을 가지고 있다는 의미이다. 6 이라면 r, w이므로 읽고 쓰기 권한만 있다.

- 파일의 소유권 변경 명령어 : chown

chown user file	file 파일의 소유자를 user로 수정
chown :group file	file 파일의 그룹을 group으로 수정
chown oracle: file	file 파일의 소유자 및 그룹을 oracle로 수정
chown user: group file	file 파일의 소유자는 user로 그룹은 group으로 수정

- 사용자 계정 개수 확인하기

```
[root@lms ~]# cat /etc/passwd | wc -l
45
```

*공부를 하다 보니 궁금점이 생겼다.
권한을 다르게 줄 필요 없이 user, group, other에게 모든 권한을 주면 편하지 않을까?
그렇다면 잘못된 명령어 사용으로 시스템 운영에 중요한 역할을 하는 파일의 내용이 변경되거나, 사용자의 사소한 실수 한 번으로 리눅스 시스템의 모든 파일이 삭제되어 버릴 수도 있기 때문에 권한을 각각 지정하는 것이다.

- 그룹 생성 및 삭제, 관리

  리눅스 사용자 계정은 어딘가의 그룹에 소속이 된다. 아무 옵션을 주지 않았다면 생성한 계정명과 동일한 이름의 그룹이 자동 생성되어 그 그룹에 속하게 된다. 그룹 관리의 핵심은 /etc/group 파일이다.

  - 사용자 그룹 확인

```
[oracle@lms ~]$ groups oracle
oracle : oinstall dba oper
```

  - 그룹 생성 (사용자 계정 생성과 마찬가지로 root에서 실행하여야 함)

```
[root@lms ~]# groupadd -r user01
[root@lms ~]# groupadd -g 1000 user02
```

옵션 X	GID는 500 부터 시작, 이전의 그룹의 GID를 1000 으로 설정하였다면 다음 그룹을 생성할 때 아무 옵션을 주지 않았을 경우 GID는 1001 이 된다.
-r	0~499 까지의 GID 중에서 가장 높은 수 다음의 수를 할당하겠다는 것
-g 1000	GID를 1000 으로 지정하는 것

  - 그룹 삭제

```
[root@lms ~]# groupdel user01
```

  - 다른 그룹으로 로그인하기

```
[root@lms ~]# groups root
root : root
[root@lms ~]# newgrp group1
[root@lms ~]# groups
group1 root
```

  작업한 내용이 로그인한 그룹명으로 기록된다.

  - 그룹 관리

```
[root@lms ~]# gpasswd group1
group1 그룹의 암호를 바꾸는 중
새 암호 :
새 암호를 다시 입력하십시오 :
```

  gpasswd 그룹명 : 그룹의 암호 설정

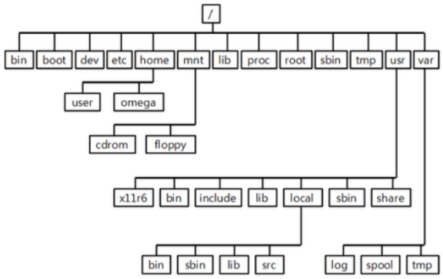
```
[root@lms ~]# gpasswd -r group1
```

  : 그룹의 암호 제거

-a	특정 그룹의 새로운 계정을 등록
-d	특정 그룹에서 지정한 계정을 삭제
-r	특정 그룹의 암호를 제거
-R	특정 그룹의 접근을 제한
-A	특정 그룹의 관리자를 설정
-M	특정 그룹의 계정을 새로 설정 (이 경우 기존 계정은 무시)

7. 디렉토리

- 디렉토리는 **파일저장소**를 의미하며 리눅스 디렉토리는 최상위 디렉토리를 기준으로 하위 디렉토리들이 존재하는 계층적 트리 구조로 구성 되어있다.



- 디렉토리의 종류와 특징

/	- 최상위 디렉토리 (루트 디렉토리) - 모든 디렉토리의 출발점
/bin	- 기본적인 명령어가 저장된 디렉토리 - mv, cp, rm 같은 기초적인 명령어들이 존재 - root 사용자와 일반 사용자가 함께 사용할 수 있음
/boot	- 리눅스 부트로더가 존재하는 디렉토리
/dev	- 시스템 디바이스 파일을 저장 - 하드디스크 장치 파일, CD-ROM 장치파일과 같은 파일 저장 - 장치들을 파일화해서 관리하며 특정 장치를 실행하기 위해선 해당 장치 파일을 실행해야 함 - 장치 파일 (device file)이나 특수 파일 (special file)은 장치 드라이버임. - 블록 장치 파일 (block devices)은 하드디스크, CD/DVD와 같은 저장 장치들이며, 문자 장치 파일 (character device)은 키보드, 마우스, 모니터, 프린터 등의 입출력 장치들이다.
/etc	- 시스템 환경 설정 파일과 부팅 관련 스크립트 파일들 저장 - /etc/sysconfig (시스템 제어판용 설정파일), /etc/pass
/home	- 사용자 계정들의 홈 디렉토리 - 일반 사용자가 로그인 시 처음으로 위치하게 되는 디렉토리
/lib	- 공유 라이브러리 디렉토리 - 커널 모듈 파일들과 프로그램 실행을 지원해 주는 라이브러리 저장 - 동적 공유 라이브러리를 저장 - 많은 프로그램에서 공통으로 사용하는 함수들을 저장하고 있어서, 디스크를 절약하며 프로그램마다 동일한 코딩을 할 필요가 없음
/mnt	- 파일 시스템을 일시적으로 마운트* 할 때 사용
/proc	- 가상파일 시스템 - 시스템 정보 디렉토리이며 커널 기능을 제어하는 역할 - 현재 메모리에 존재하는 모든 작업(프로세스)들이 파일형태로 존재

/root	- 시스템 관리자용 홈 디렉토리
/sbin	- 관리자용 시스템 표준 명령 및 시스템 관리와 관련된 실행 명령어 (ifconfig, e2fsck, ethtool, halt 등) 저장
/tmp	- 각종 프로그램이나 프로세스 작업을 할 때 임시로 생성되는 파일 저장 - 공용 디렉토리, 모든 사용자에 대해 읽기 쓰기 허용 - 스티키 비트(Sticky bit)* 설정으로 파일의 소유자만이 자신의 파일을 지울 수 있음
/usr	- 일반 사용자 디렉토리로 사용자 데이터나 어플리케이션 저장
/var	- 가변 자료 저장 디렉토리로 로그 파일이나 메일 데이터 저장
/lost+found	- 결함이 있는 파일에 대한 정보가 저장
/proc	- 사용자가 /proc이나 그 하위 파일에 접근할 때마다 커널에서 파일 내용을 동적으로 만들어 냄 - 각 프로세서는 고유의 식별자를 가지고 있으며, 이 식별자를 가진 디렉토리 하위에 정보를 저장

*마운트 : 디스크와 같은 물리적인 장치를 디렉터리에 연결시켜주는 것