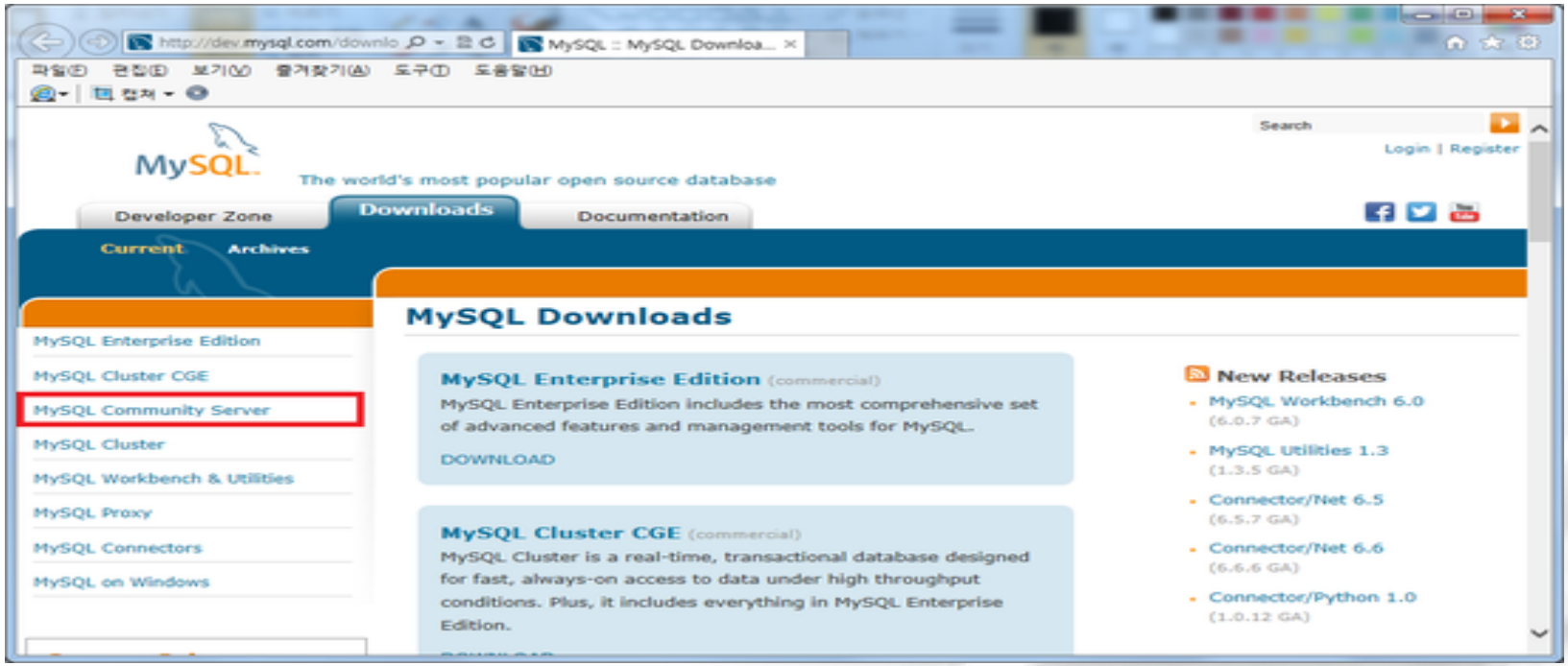
MySQL

1. MySQL

- SQL에 기반을 둔 관계형 DBMS 중 하나
- 리눅스, 유닉스, 윈도우 등 거의 모든 운영체제에서 사용 가능
- 리눅스에서 사용하는 표준 데이터베이스였지만 지금은 MySQL이 오라클에 인수됨으로 인해 동일한 엔진으로 만들어져 있는 Maria DB가 리눅스의 표준 데이터베이스
- 처리 속도가 빠르고 대용량 데이터 처리 용이
- 설치 방법이 쉽고 초보자도 익히기 쉬움
- 보안성이 우수
- Standard, Enterprise, Cluster CGE 3개의 상용 버전과 Commuity 무료 버전이 제공

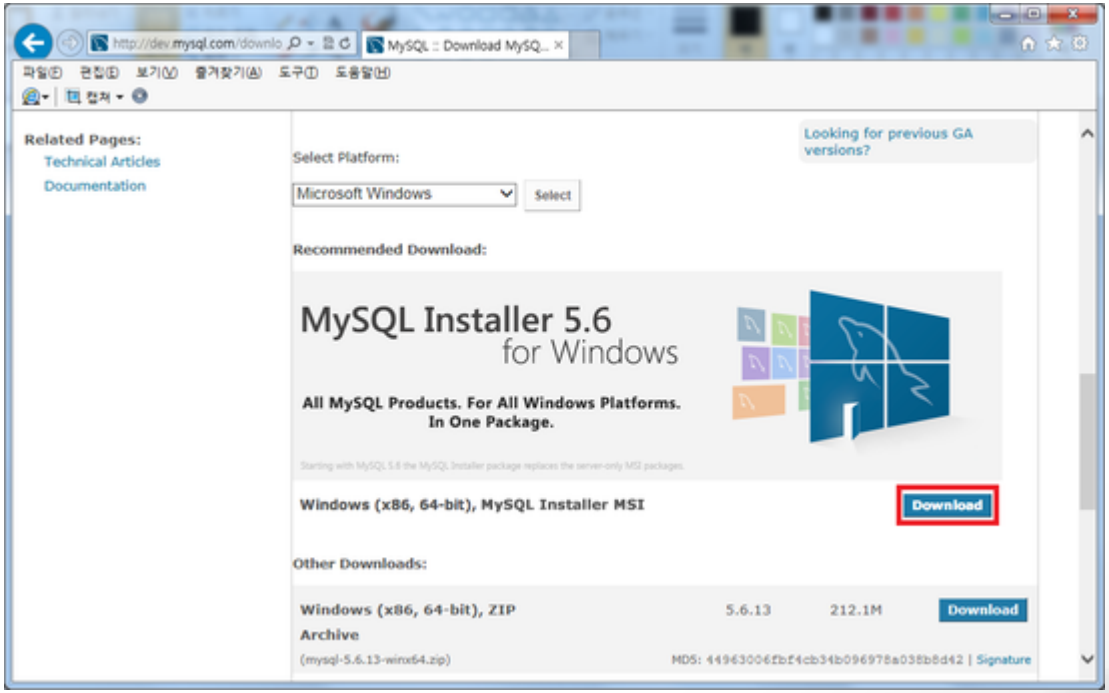
2. MySQL 설치

- 오라클 사이트에 접속해서 다운로드
- 아래 주소로 직접 이동해서 다운로드 가능
  https://dev.mysql.com/downloads/mysql/사이트에서 다운로드



2. MySQL 설치

- Windows 의 MSI 버전은 닷넷 프레임워크 4.0 이상과 Visual C++ 2013 재배포 패키지가 설치되어 있어야 함



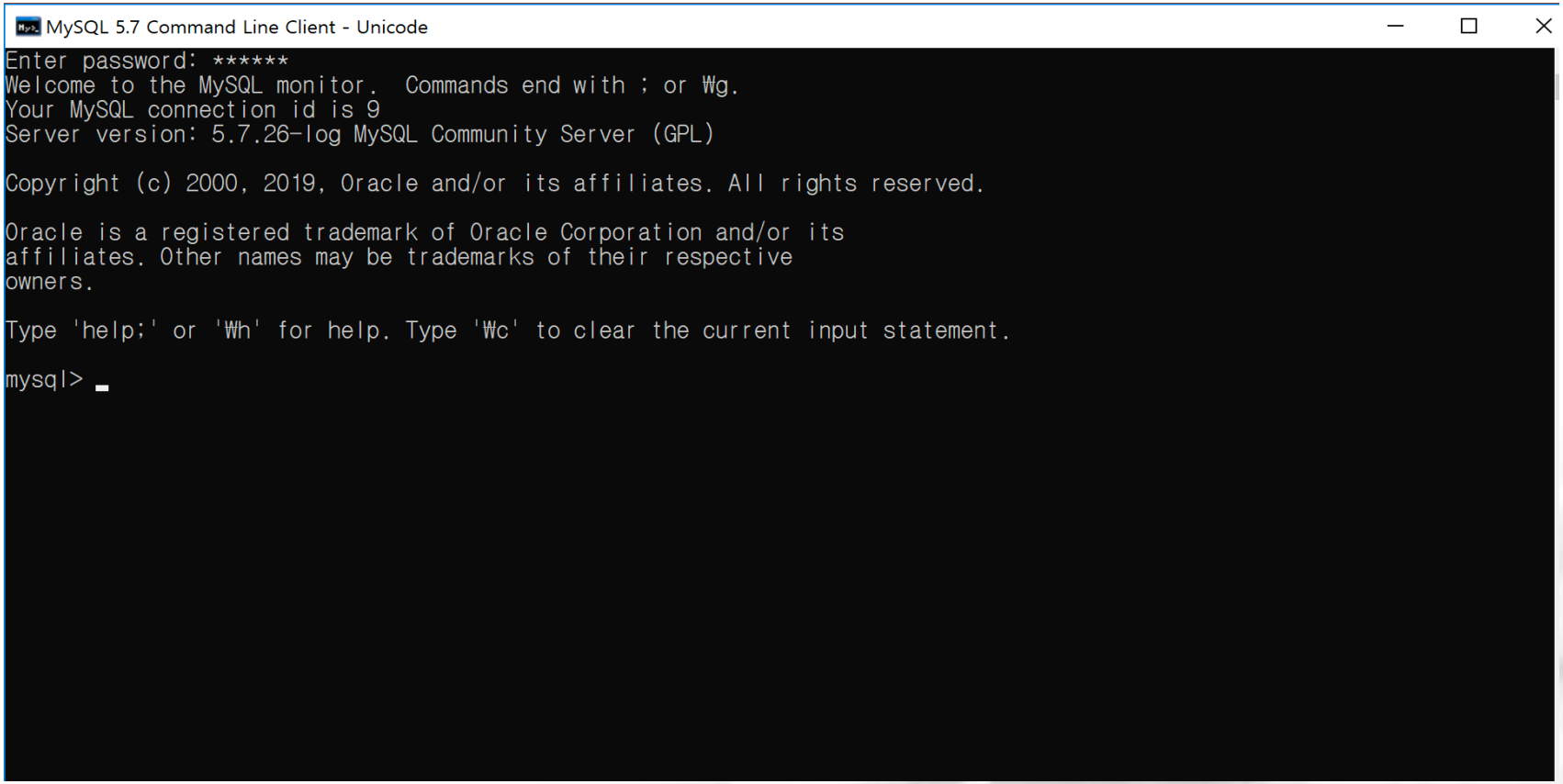
2. MySQL 설치

❖ Windows – 다운받은 설치파일을 실행해서 Install MySQL Products 를 선택

2. MySQL 설치

❖ 설치 확인

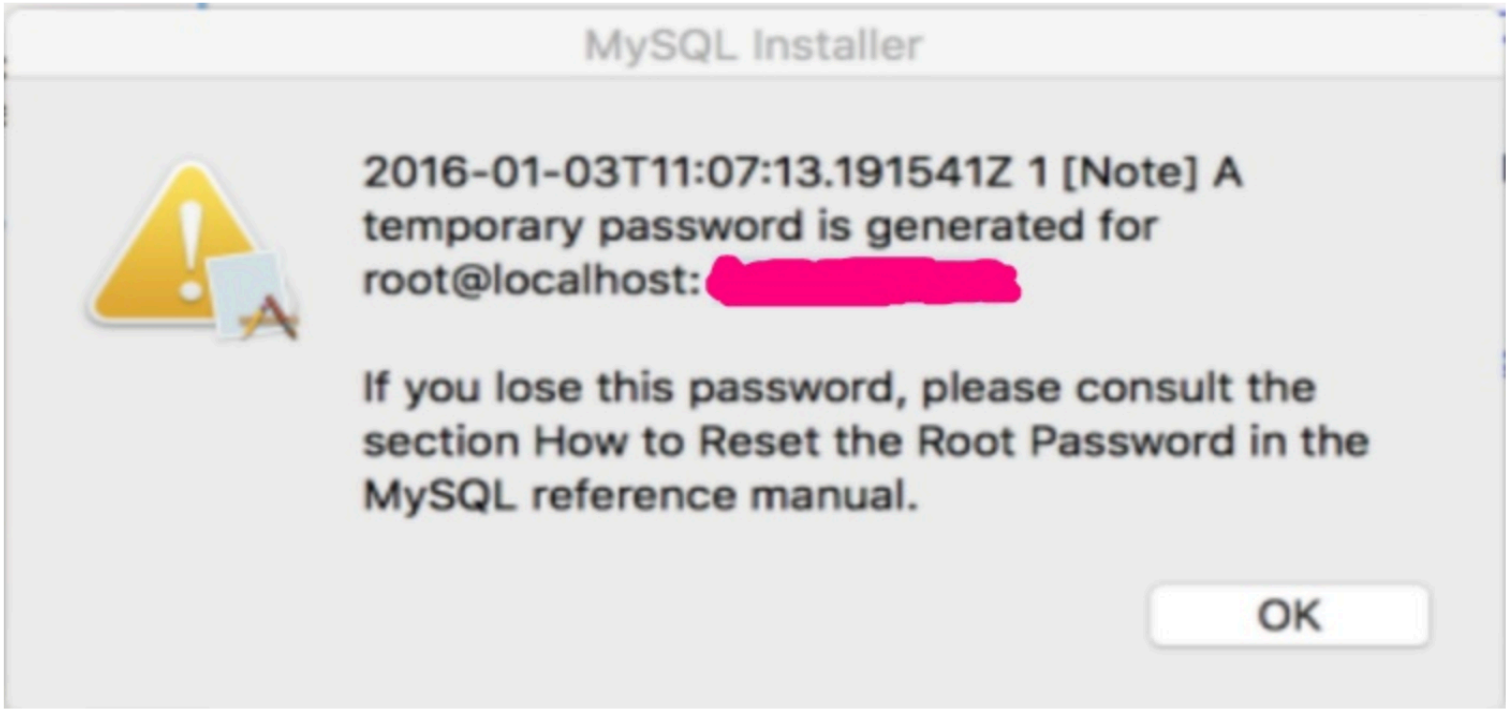
✓ MySQL Command Line Client 를 실행시켜서 설정한 관리자 비밀번호를 입력해서 접속



2. MySQL 설치

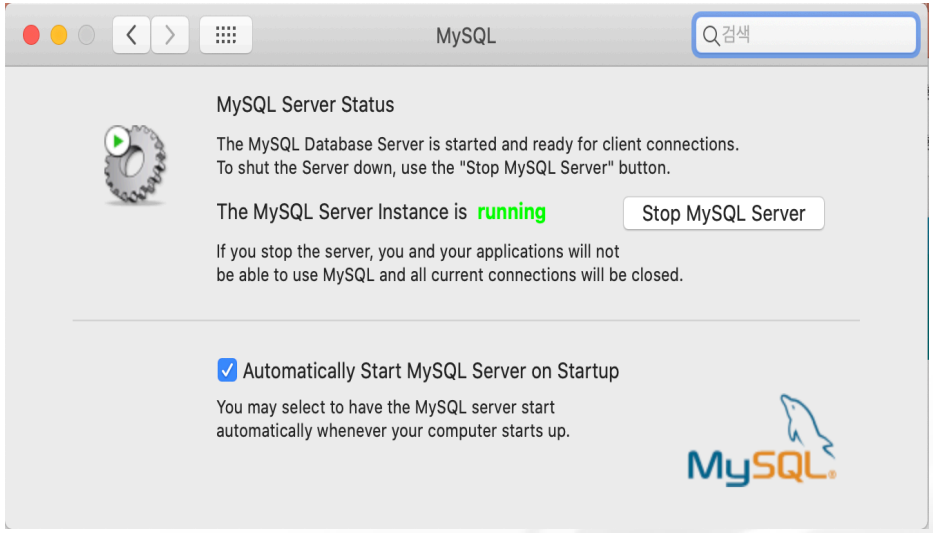
- Mac에서는 다운로드 받을 때나 설치 중에 아래와 같은 창이 출력되는데 이 창에 초기 root 비밀번호를 출력해기 때문에 기록을 해 두어야 함

주의



2. MySQL 설치

- ❖ Mac에 설치 한 경우 root 비밀번호 변경
  - ✓ 서비스를 실행해야 함 – 환경설정

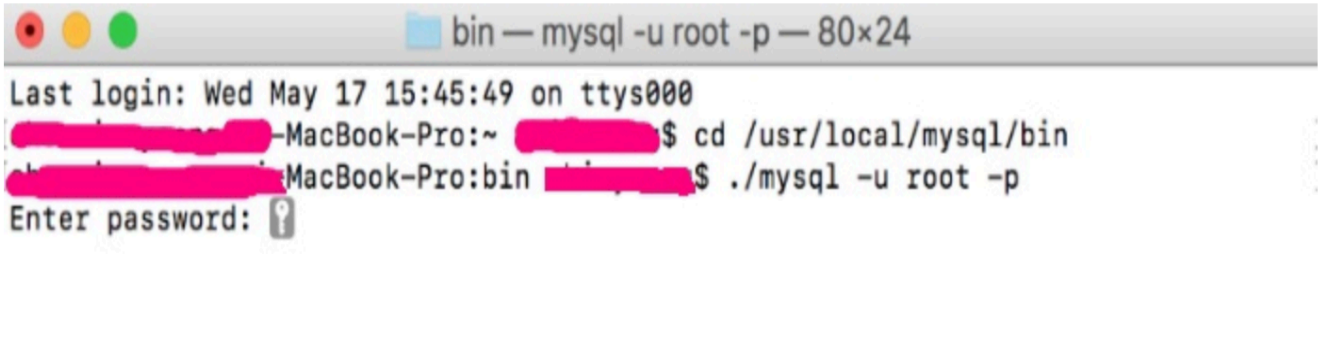


2. MySQL 설치

- Mac에 설치 한 경우 root 비밀번호 변경
  - ✓ 터미널에서 아래 명령을 수행하고 임시 비밀번호를 입력

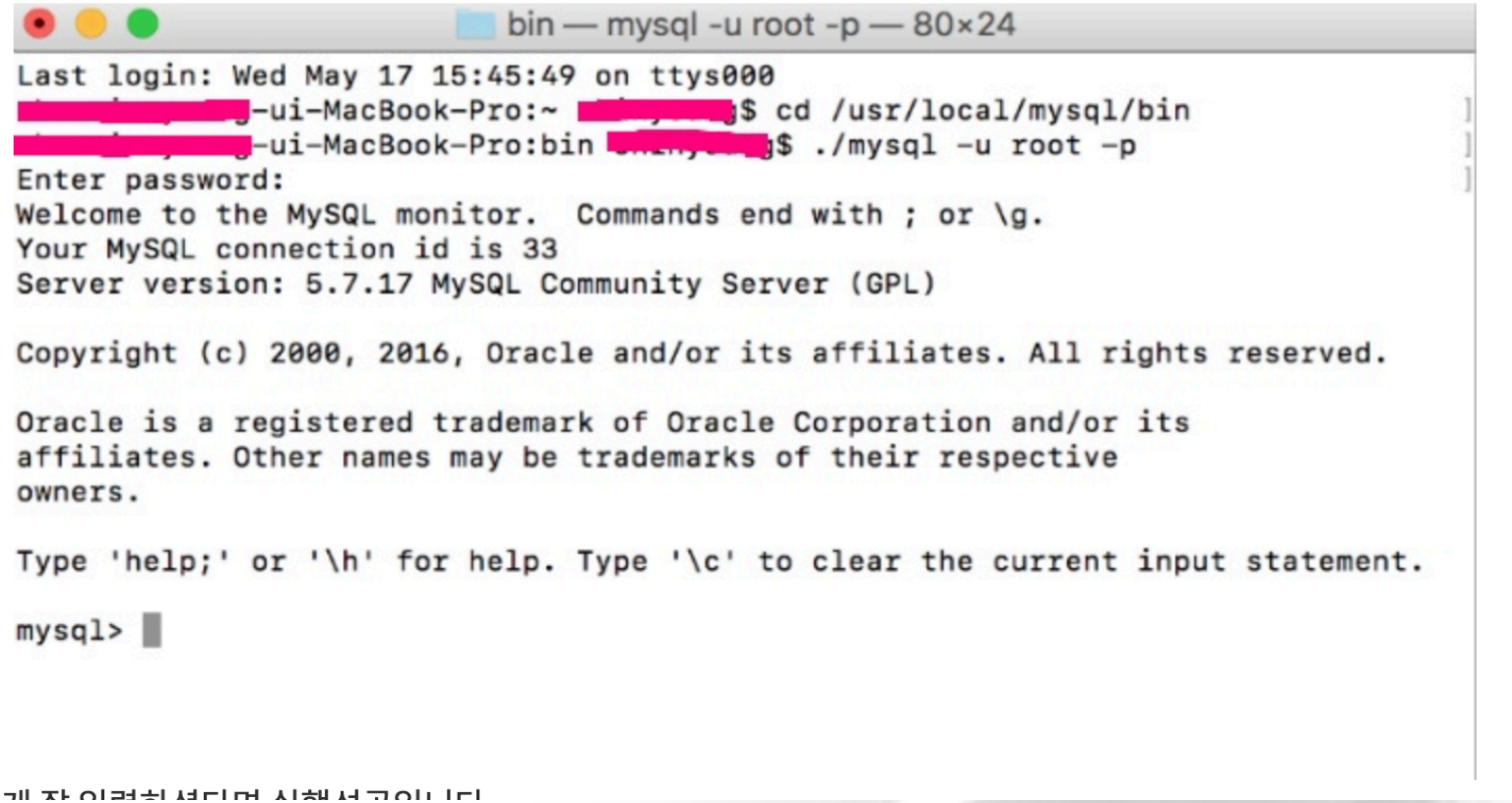
터미널에 아래와 같이 입력합니다.

```
1 cd /usr/local/mysql/bin
2 ./mysql -u root -p
```



2. MySQL 설치

❖ Mac에 설치 한 경우 root 비밀번호 변경



2. MySQL 설치

❖ Mac에 설치 한 경우 root 비밀번호 변경

```
mysql> use mysql;

mysql> update user set authentication_string=password('root') where user='root';

mysql> flush privileges;
```

2. MySQL 설치

❖ Mac에서 MySQL을 삭제할 때는 터미널에서 아래 명령을 순서대로 입력

1. sudo rm /usr/local/mysql
2. sudo rm -rf /usr/local/mysql*
3. sudo rm -rf /Library/StartupItems/MySQLCOM
4. sudo rm -rf /Library/PreferencePanes/My*
5. sudo rm -rf /Library/LaunchDaemons/com.microsoft.office.licensing.helper.plist
6. sudo rm -rf /private/var/db/receipts/*mysql*
7. rm -rf ~/Library/PreferencePanes/My*
8. sudo rm -rf /Library/Receipts/mysql*
9. sudo rm -rf /Library/Receipts/MySQL*
10. sudo rm -rf /var/db/receipts/com.mysql.*

3. MySQL 접속

MySQL 접속 명령 1

```
C:₩> mysql -u계정 -p비밀번호
mysql> use 데이터베이스명
```

MySQL 접속 명령 2

```
C:₩> mysql -u계정 -p비밀번호 데이터베이스명
```

ex)

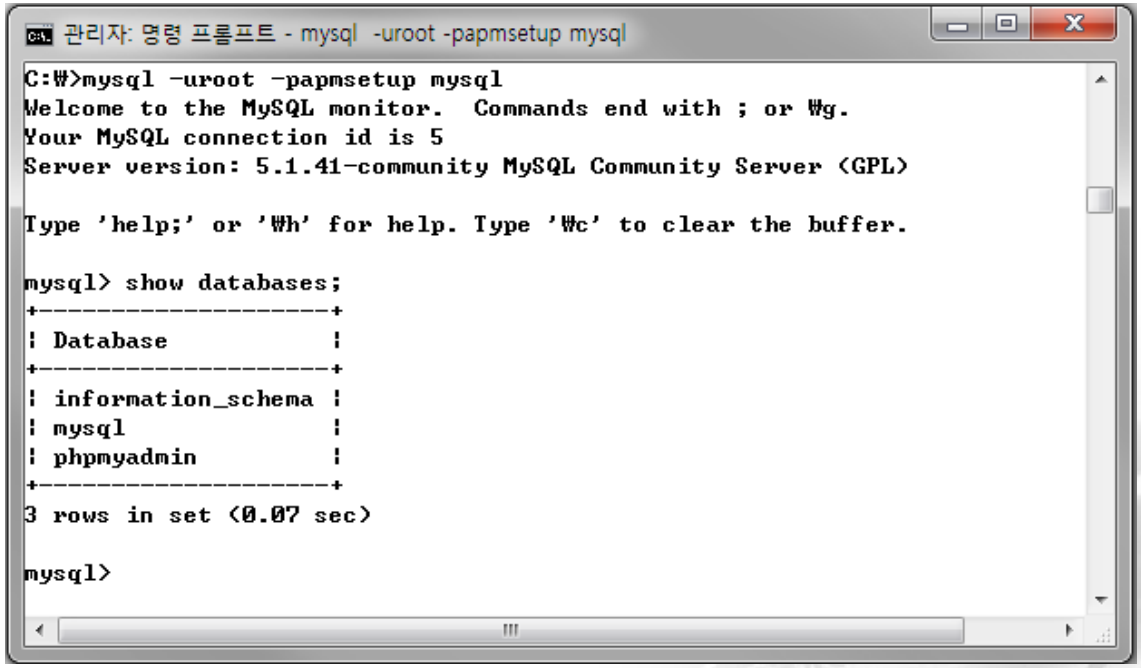
```
계정 : root, 비밀번호:1234, DB명:mysql
 es:₩> mysql -uroot -p1234 mysql
```

3. MySQL 접속

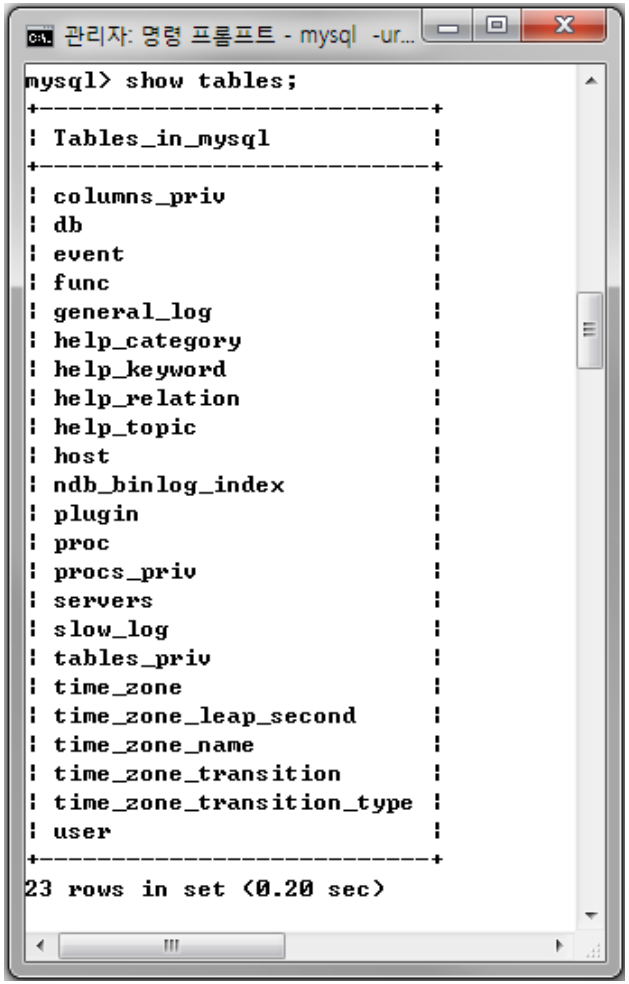
❖ MySQL 접속 종료

3. MySQL 접속

- 데이터베이스에 관리자 계정으로 접속
- 존재하는 데이터베이스 목록보기
  mysql> show databases;
- 데이터베이스 사용
  mysql> use 데이터베이스이름;



3. MySQL 접속



❖ 테이블 목록보기

mysql> show tables;

3. MySQL 접속

테이블 구조 출력 명령

```
mysql> desc 테이블명;
```

4. 데이베이스 접속 권한 설정

grant all privileges on 데이터베이스이름.* to 계정@'%' identified by '비밀번호'

- 데이터베이스 이름 대신에 *을 대입하면 모든 데이터베이스 사용 가능
- %대신에 ip를 기재하면 특정 ip에서만 접속이 허용되며 localhost로 지정하면 현재 컴퓨터에서만 접속이 가능

5. 데이터베이스 관련 명령

데이터베이스 생성 명령

```
create database 데이터베이스명;
```

데이터베이스 목록 출력 명령

```
show databases;
```

데이터베이스 삭제 명령

```
drop database 데이터베이스명;
```

데이터베이스 사용 명령

```
use 데이터베이스명;
```

6. 테이블 관련 명령

데이터베이스 테이블 생성 명령

```
create [temporary] table [if not exists]테이블명(
            컬럼명1 타입 [Constaraint 제약조건이름] 컬럼제약조건,
            컬럼명2 타입,
            컬럼명3 타입,
            ..........
             [Constaraint 제약조건이름] 테이블 제약 조건)ENGINE=엔진명;
```

6. 테이블 관련 명령

❖ MySQL에서 지원하는 데이터 형식의 종류
  ✓ 숫자 데이터 형식

데이터 형식	바이트 수	숫자 범위	설명
BIT(N)	N/8		1~64bit를 표현. b'0000' 형식으로 표현
TINYINT	1	-128~127	정수
★SMALLINT	2	-32,768~32,767	정수
MEDIUMINT	3	-8,388,608~8,388,607	정수
★INT INTEGER	4	약 -21억~+21억	정수
★BIGINT	8	약 -900경~+900경	정수
★FLOAT	4	-3.40E+38~-1.17E-38	소수점 아래 7자리까지 표현
★DOUBLE REAL	8	-1.22E-308~1.79E+308	소수점 아래 15자리까지 표현
★DECIMAL(m, [d]) NUMERIC(m, [d])	5~17	$-10^{38}+1$~$+10^{38}-1$	전체 자릿수(m)와 소수점 이하 자릿수(d)를 가진 숫자형 예) decimal(5, 2)은 전체 자릿수를 5자리로 하되, 그 중 소수점 이하를 2자리로 하겠다는 의미

6. 테이블 관련 명령

❖ MySQL에서 지원하는 데이터 형식의 종류
  ✓ 문자 데이터 형식

데이터 형식		바이트 수	설명
★CHAR(n)		1~255	고정길이 문자형. n을 1부터 255까지 지정. character의 약자 그냥 CHAR만 쓰면 CHAR(1)과 동일
★VARCHAR(n)		1~65535	가변길이 문자형. n을 사용하면 1부터 65535까지 지정. Variable character의 약자
BINARY(n)		1~255	고정길이의 이진 데이터 값
VARBINARY(n)		1~255	가변길이의 이진 데이터 값
TEXT 형식	TINYTEXT	1~255	255 크기의 TEXT 데이터 값
	TEXT	1~65535	N 크기의 TEXT 데이터 값
	MEDIUMTEXT	1~16777215	16777215 크기의 TEXT 데이터 값
	★LONGTEXT	1~4294967295	최대 4GB 크기의 TEXT 데이터 값
BLOB 형식	TINYBLOB	1~255	255 크기의 BLOB 데이터 값
	BLOB	1~65535	N 크기의 BLOB 데이터 값
	MEDIUMBLOB	1~16777215	16777215 크기의 BLOB 데이터 값
	★LONGBLOB	1~4294967295	최대 4GB 크기의 BLOB 데이터 값
ENUM(값들…)		1 또는 2	최대 65535개의 열거형 데이터 값
SET(값들…)		1, 2, 3, 4, 8	최대 64개의 서로 다른 데이터 값

6. 테이블 관련 명령

❖ MySQL에서 지원하는 데이터 형식의 종류
- ✓ BOOL: true 또는 false
- ✓ JSON: 5.7.8 버전 이후에서 제공
- ✓ 날짜 데이터 형식

DATE	1000-01-01 ~ 9999-12-31	YYYY-MM-DD
DATETIME	1000-01-01 00:00:00 ~ 9999-12-31 23:59:59	YYYY-MM-DD HH:MM:SS
TIMESTAMP	1970-01-01 ~ 2037년 임의 시간(1970-01-01 00:00:00 를 0으로 해서 1초단위로 표기)	
TIME	-838:59:59 ~ 838:59:59	
YEAR	901~2155	

6. 테이블 관련 명령

❖제약조건

- ✓ unique: 유일성
- ✓ primary key: 기본키
- ✓ not null: 필수
- ✓ check: 값의 목록이나 범위를 설정
- ✓ foreign key: 참조키, 외래키

  foreign key(컬럼이름) refernces 다른테이블(컬럼이름)
- ✓ auto_increment: 자동으로 1부터 증가하는 값을 입력해 주는데 숫자 형식에만 사용할 수 있고 primary key나 unique와 함께 설정
- ✓ default: 기본값

6. 테이블 관련 명령

❖ENGINE

- ✓ MyISAM
  - 조회에 유리
  - KEY INDEX를 지원하여 KEY로 정의된 칼럼의 값을 조회 조건으로 넣으면 INDEX로 빠르게 검색하는 기능을 소유
- ✓ InnoDB
  - 트랜잭션 처리에 유리하지만 KEY를 이용한 검색에는 불리

❖DEFAULT CHARSET 인코딩방식 을 이용해서 인코딩 방식 설정 가능

❖테이블 생성 시 auto_increment=초기값 을 추가하면 시퀀스의 초기 값을 설정할 수 있음

❖ALTER TABLE [테이블명] auto_increment=[시작하려는 값] 을 이용해서 초기값을 재설정 할 수 있음

6. 테이블 관련 명령

연락처 테이블(테이블명: contact)

컬럼명	타입	설명
num	int	일련번호, 기본키
name	char(20)	이름
address	varchar(100)	주소
tel	char(20)	전화번호
email	char(100)	이메일 주소
birthday	date	생일

6. 테이블 관련 명령

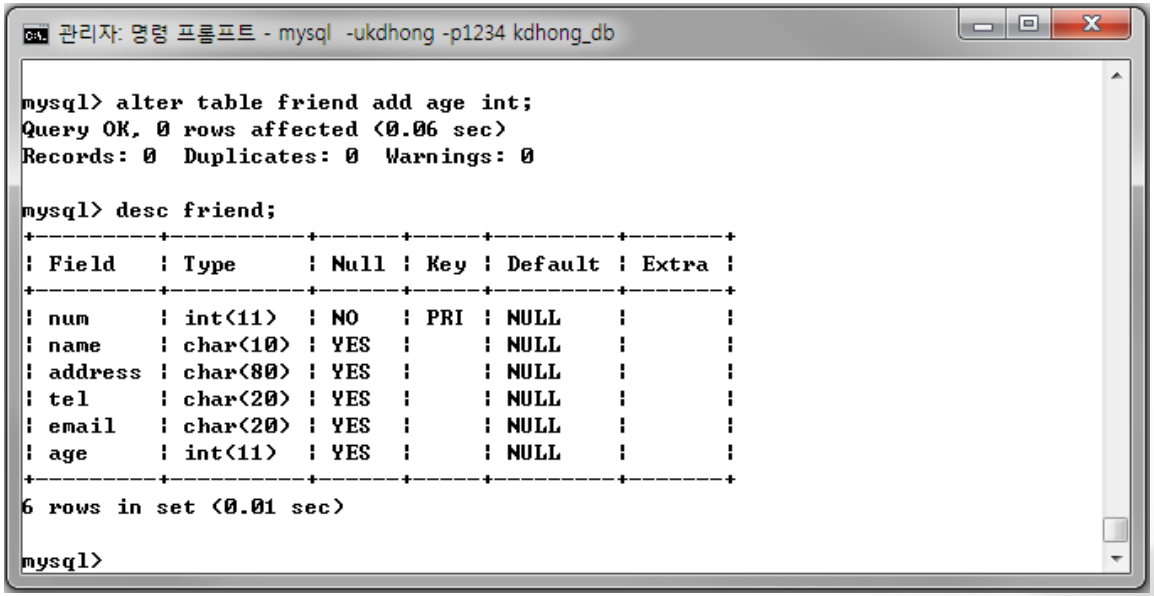
❖ 연락처 테이블(contact) 만들기

```
create table contact (
num integer auto_increment primary key,
name char(20),
address varchar(100),
tel char(20),
email char(100),
birthday date
) ENGINE=MyISAM auto_increment=1 DEFAULT CHARSET=utf8;
```

6. 테이블 관련 명령

데이터베이스 테이블의 컬럼 추가 명령

alter table 테이블명 add 새로운 컬럼명 컬럼타입 [first 또는 after 컬럼명];



새로운 컬럼 추가 명령

ex)앞서 만든 contact 테이블에 나이 컬럼을 정수형으로 추가

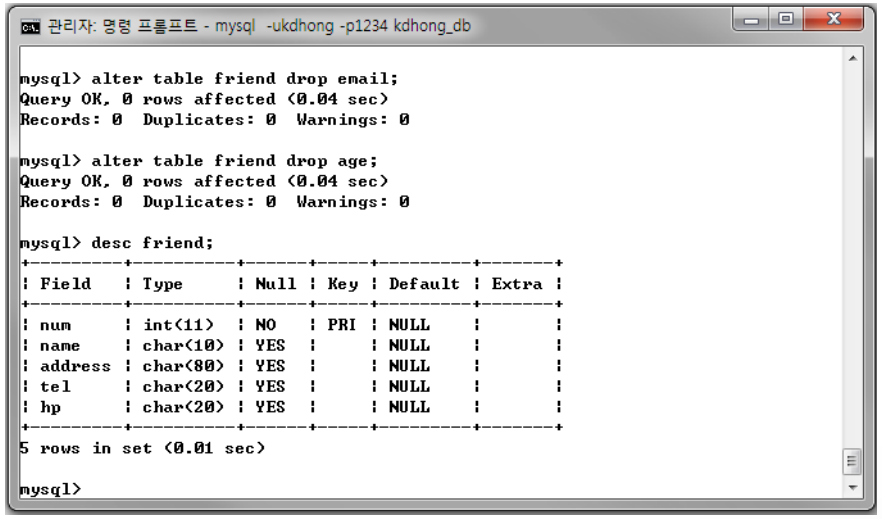
alter table contact add age int;

desc contact;

6. 테이블 관련 명령

데이터베이스 테이블의 컬럼 삭제 명령

alter table 테이블명 drop 삭제할 컬럼명1, 삭제할 컬럼명2;



컬럼 삭제 – 제약조건이 설정된 경우 제약조건을 먼저 삭제

ex) contact 테이블에서 age 컬럼 삭제

```
alter table contact drop age;

desc contact;
```

6. 테이블 관련 명령

데이터베이스 테이블의 컬럼 수정 명령

```
alter table 테이블명 change 이전 컬럼명 새로운 컬럼명 컬럼 타입;
```

테이블 컬럼 타입 수정 명령

```
alter table 테이블명 modify 컬럼명 새로운 타입;
```

컬럼 수정 명령
ex) contact 테이블 컬럼 중 tel char(20)을 phone int로 변경
alter table contact change tel phone int;

6. 테이블 관련 명령

테이블 제약조건 수정

```
alter table 테이블이름 MODIFY 컬럼이름 자료형 제약조건;
```

테이블 제약조건 추가

```
alter table 테이블이름 ADD 제약조건(컬럼이름)
```

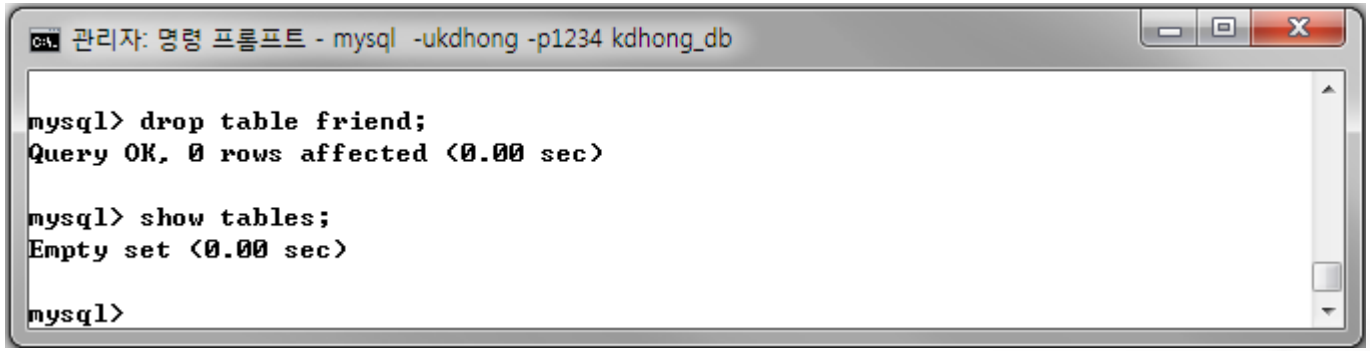
테이블 제약조건 삭제

```
alter table 테이블이름 DROP CONSTRAINT 제약조건이름
```

6. 테이블 관련 명령

데이터베이스 테이블 삭제 명령

drop table 테이블명;



테이블 삭제

ex) contact 테이블 삭제
drop table contact;

show tables;

6. 테이블 관련 명령

데이터베이스 테이블의 전체 데이터 삭제 명령

```
truncate table 테이블명;
```

테이블 이름 수정 명령

```
alter table 이전 테이블명 rename 새로운 테이블명;
```

7. 레코드 관련 명령

테이블에 저장된 모든 데이터 조회

```
select *
from 테이블이름
```

usertbl 테이블과 buytbl 테이블의 모든 데이터 조회하기

7. 레코드 관련 명령

테이블에서 특정 컬럼 만 출력하기

```
select 컬럼명1, 컬럼명2…
from 테이블명;
```

조건에 맞는 레코드 검색 명령

```
select 컬럼명1, 컬럼명2
from 테이블명
where 조건식;
```

조건에 사용 가능한 연산자

>, >=, <, <=, =, !=, between A and B, in, not in, is null, is not null

7. 레코드 관련 명령

usertbl 테이블에서 name 이 '김태연' 인 데이터 조회

usertbl 테이블에서 birthyear가 1990년 이후 이고 addr 이 '서울' 인 데이터를 조회

usertbl 테이블에서 birthyear가 1990년 이후 이거나 addr 이 '서울' 인 데이터를 조회

usertbl 테이블에서 birthyear가 1990에서 1993 사이인 데이터 조회

7. 레코드 관련 명령

특정 문자열이 포함된 레코드 검색 명령

```
select 컬럼명1, 컬럼명2
from 테이블명
where 검색 컬럼 like 조건식;
```

%: 0개 이상의 문자열
_: 1개의 문자

%와 _를 검색하고자 하는 경우에는 ₩를 앞에 추가
‘를 검색어에 포함시켜서 검색하고자 하는 경우에는 ‘’를 기재

usertbl 테이블에서 name에 “라”가 포함된 데이터의 모든 컬럼을 조회

usertbl 테이블에서 name이 “배”로 시작하는 데이터의 모든 컬럼을 조회

7. 레코드 관련 명령

단일 행 서브쿼리

- 쿼리문 안에 또 쿼리문이 들어 있는 것
- 태연보다 태어난 해가 작거나 같은 경우

where 조건에 태연의 태어난 해를 직접 써줘야 함

```
select * from usertbl where birthyear >= 1989;

select * from usertbl
where birthyear >=
            (select birthyear from usertbl where Name = '김태연');
```

7. 레코드 관련 명령

다중 행 서브쿼리

- 서브쿼리의 결과가 하나의 행이 아니고 여러 행 일 때는 ANY, ALL, SOME, IN, NOT IN을 이용해서 비교
- 서브쿼리 앞에 EXISTS, NOT EXISTS를 추가해서 데이터가 존재하는 경우와 그렇지 않은 경우에 작업을 다르게 할 수 있음
- buytbl 테이블에서 userid가 kty 인 데이터가 구매한 productname 과 동일한 제품을 구매한 userid를 조회

7. 레코드 관련 명령

레코드 개수 설정 - 페이징

select 컬럼명1, 컬럼명2 from 테이블명 where 조건식 limit 개수;
select 컬럼명1, 컬럼명2 from 테이블명 where 조건식 limit 시작위치, 개수;

- ✓ 표준은 아니라서 MySQL 이나 PostgreSQL에서 사용 가능하며 가장 마지막 절에 설정
- ✓ 시작위치는 0부터 시작

usertbl 테이블에서 birthyear가 1990 보다 작은 데이터 2개를 조회

중복 데이터 제거 - distinct

select 절에서 컬럼 이름 앞에 distinct를 기재하면 중복 데이터가 제거

7. 레코드 관련 명령

데이터 그룹화

```
select * 또는 컬럼이름 나열
from 테이블이름
where 조건
group by 그룹화할 연산식이나 컬럼이름
```

집계함수

- ✓ SUM()
- ✓ AVG()
- ✓ MIN()
- ✓ MAX()
- ✓ COUNT() : 다른 집계 함수는 컬럼이름이나 연산식을 매개변수로 사용하지만 COUNT는 종종 *을 매개변수로 사용
- ✓ STDDEV
- ✓ VAR_SAMP()

7. 레코드 관련 명령

- ✓ buytbl 테이블의 데이터 개수 조회
- ✓ buytbl에서 평균 구매 개수 조회
- ✓ buytbl에서 구매자 별 평균 구매 개수 조회

7. 레코드 관련 명령

데이터 그룹화 한 후 조건

```
select * 또는 컬럼이름 나열
from 테이블이름
where 조건
group by 그룹화할 연산식이나 컬럼이름
having 그룹화 한 이후의 조건
```

✓ 구매횟수가 3번 이상인 사람의 아이디 조회

7. 레코드 관련 명령

레코드 정렬 명령

select 컬럼명1, 컬럼명2
from 테이블 명
order by 컬럼명이나 표현식[ASC | DESC] …;

- ✓ 기본은 오름차순이고 desc를 추가하면 내림차순 정렬을 수행
- ✓ 컬럼 이름은 여러 개 나열이 가능한데 앞 컬럼의 값이 동일할 때 두번째 컬럼을 가지고 정렬
- ✓ NULL은 가장 작은 값으로 간주

usertbl 테이블의 데이터를 birthyear의 오름차순으로 출력

usertbl 테이블의 데이터를 birthyear의 오름차순으로 출력하고 동일한 데이터의 경우는 name의 내림차순으로 정렬

7. 레코드 관련 명령

열이나 테이블에 별명 사용

- 열이름나 연산식 뒤에 AS 를 추가하고 별명을 입력할 수 있으며 AS는 생략이 가능하고 한글을 사용하는 경우에는 “ ”로 묶어 주어야 함
- from 절에서는 테이블이름 뒤에 별명을 입력하면 되는데 이 경우에는 이후에는 테이블이름 대신에 별명을 사용

7. 레코드 관련 명령

ROUND 함수

❖ ROUND(컬럼이름이나 표현식, 반올림할 자릿수)

✓ 소수 자릿수는 양수로 설정하면 되고 음수는 1의 자리부터 반올림

buytbl 테이블에서 userid 별로 amount 의 평균을 조회하는데 소수 첫째 자리에서 반올림해서 출력

7. 레코드 관련 명령

문자열 함수

- 문자열 결합은 concat 함수를 이용
- 문자열의 추출은 substring(컬럼이름, 시작위치, 추출할개수)
- 좌우공백 제거는 trim
- 문자의 개수는 char_length
- 바이트의 개수는 octet_length

usertbl 테이블의 name 과 birthyear를 결합해서 조회

7. 레코드 관련 명령

날짜 연산

- 현재 날짜 및 시간
  - ✓ CURRENT_DATE() -> 현재 날짜
  - ✓ CURRENT_TIME() -> 현재 시간
  - ✓ CURRENT_TIMESTAMP() -> 현재 날짜 및 시간

- 특정 날짜 생성
  - ✓ str_to_date(날짜문자열, 날짜 서식 문자열)
  - ✓ select STR_TO_DATE('1986-05-05 11:00:00', '%Y-%m-%d %H:%i:%S') ;

- 날짜와 기간형 데이터와 연산
  - ✓ 날짜 데이터 와 INTERVAL 정수 단위 형태의 덧셈과 뺄셈 가능
  - ✓ CURRENT_DATE() + INTERVAL 1 DAY : 현재 날짜에 하루를 추가

- 날짜 간의 뺄셈
  - ✓ DATEDIFF(날짜 데이터, 날짜 데이터)
  - ✓ select datediff(CURRENT_DATE(), STR_TO_DATE('1986-05-05', '%Y-%m-%d')) ;

7. 레코드 관련 명령

레코드 삽입 명령

INSERT INTO 테이블명(컬럼명 나열) VALUES(값 나열)

- ✓ 모든 컬럼에 순서대로 값을 대입할 때는 컬럼명 생략 가능
- ✓ auto_increment는 값을 대입할 필요가 없음
- ✓ 테이블에 존재하는 컬럼 생략된 컬럼은 DEFAULT가 있는 경우에는 DEFAULT 값이 저장되고 그렇지 않은 경우는 NULL
- ✓ NULL은 명시적으로 NULL이라고 입력 가능
- ✓ 기본값을 설정할 때도 명시적으로 DEFAULT 사용 가능

usertbl 테이블에 userid가 kjn, name은 제니, birthyear는 1996 그리고 addr은 서울, mobile은 01012341234 그리고 mdate는 1996년 1월 16일 인 데이터를 추가

7. 레코드 관련 명령

레코드 수정 명령

UPDATE 테이블명
set 컬럼명=컬럼값
[where 조건식]

usertbl 테이블에서 userid 가 kjn 인 데이터의 name을 김제니로 수정

조건식을 만족하는 특정 레코드 삭제 명령

DELETE from 테이블명
[where 조건식]

usertbl 테이블에서 userid 가 kjn 인 데이터를 삭제

8. JOIN

❖ CROSS JOIN(상호 조인)
- ✓ 한쪽 테이블의 모든 행들과 다른 쪽 테이블의 모든 행 조인
- ✓ CROSS JOIN의 결과 개수는 두 테이블 개수를 곱한 개수
- ✓ 카티션 곱(Cartesian Product) 이라고도 부름
- ✓ from 절에 2개의 테이블 이름이 존재하는 경우

회원 테이블(userTbl)

아이디	이름	생년	지역	국번	전화번호	키	가입일
LSG	이승기	1987	서울	011	1111111	182	2008.8.8
KBS	김범수	1979	경남	011	2222222	173	2012.4.4
KKH	김경호	1971	전남	019	3333333	177	2007.7.7
JYP	조용필	1950	경기	011	4444444	166	2009.4.4
SSK	성시경	1979	서울			186	2013.12.12
LJB	임재범	1963	서울	016	6666666	182	2009.9.9
YJS	윤종신	1969	경남			170	2005.5.5
EJW	은지원	1978	경북	011	8888888	174	2014.3.3
JKW	조관우	1965	경기	018	9999999	172	2010.10.10
BBK	바비킴	1973	서울	010	0000000	176	2013.5.5

PK

구매 테이블(buyTbl)

순번	아이디	물품명	분류	단가	수량
1	KBS	운동화		30	2
2	KBS	노트북	전자	1000	1
3	JYP	모니터	전자	200	1
4	BBK	모니터	전자	200	5
5	KBS	청바지	의류	50	3
6	BBK	메모리	전자	80	10
7	SSK	책	서적	15	5
8	EJW	책	서적	15	2
9	EJW	청바지	의류	50	1
10	BBK	운동화		30	2
11	EJW	책	서적	15	1
12	BBK	운동화		30	2

PK FK

8. JOIN

❖ CROSS JOIN(상호 조인)

```
select *
from usertbl, buytbl;

select *
from usertbl cross join buytbl;
```

✓ 위 구문의 결과는 컬럼의 개수는 2개 테이블의 컬럼의 합이 되고 행의 개수는 2개 테이블의 행의 개수 곱

8. JOIN

❖ INNER JOIN(내부 조인)

- ✓ 2개의 테이블에 공통된 의미를 갖는 컬럼이 존재하는 경우에 수행
- ✓ 2개 컬럼의 공통된 데이터만 결합 – EQUI JOIN
- ✓ = 이외의 <>, >, >=, <, <= 연산자를 이용 – NON EQUI JOIN
- ✓ 사용 형식

```
SELECT <열 목록>
FROM <첫 번째 테이블>
        INNER JOIN <두 번째 테이블>
        ON <조인될 조건>
[WHERE 검색조건]
```

- ✓ usertbl 테이블과 buytbl에 공통으로 존재하는 컬럼의 값을 가지고 where 절에서 비교해도 되는데 이 경우에는 양쪽 테이블의 컬럼이름이 같은 경우에는 앞에 테이블이름.을 추가해서 표현
- ✓ 2개 테이블의 컬럼 이름이 동일한 경우 inner join 대신에 natural join으로 입력하면 조인 조건을 작성할 필요가 없음

8. JOIN

❖ INNER JOIN(내부 조인)

✓ usertbl 테이블과 buytbl에는 userid 가 공통으로 존재 – inner join 수행

```
select *
from usertbl, buytbl
where usertbl.userid  = buytbl.userid;

select *
from usertbl inner join buytbl on usertbl.userid = buytbl .userid;

select *
from usertbl natural join buytbl;
```

8. JOIN

❖ OUTER JOIN(외부 조인)

✓ 한쪽에만 존재하는 행까지도 포함시키는 것

```
SELECT <열 목록>
FROM <첫 번째 테이블(LEFT 테이블)>
 <LEFT ¦ RIGHT ¦ FULL> OUTER JOIN <두 번째 테이블(RIGHT 테이블)>
      ON <조인될 조건>
[WHERE 검색조건] ;
```

✓ usertbl 테이블에는 존재하지만 buytbl에는 존재하지 않은 산다라박 과 김설현도 조회

```
select *
from usertbl left outer join buytbl on usertbl.userid = buytbl .userid;
```

8. JOIN

- SELF JOIN(자체 조인)
  - ✓ 자기 자신과 자기 자신이 조인하는 경우
  - ✓ 동일한 의미를 갖는 컬럼이 하나의 테이블에 중복해서 존재하는 경우 가능

8. JOIN

❖ 집합 연산
- ✓ 동일한 구조를 갖는 테이블끼리의 연산
- ✓ 구조가 동일하지 않으면 집합 연산은 수행할 수 없음
- ✓ 형식

select 구문
UNION
select 구문
[order by 정렬할 컬럼이름 또는 표현식]

- ✓ UNION 대신에 UNION ALL을 이용하면 중복된 데이터를 제거하지 않음
- ✓ 공통된 요소만 추출하고자 하는 경우에는 UNION 대신에 INTERSECT 한 쪽에만 존재하는 데이터를 추출하고자 하는 경우에는 EXCEPT(Oracle 은 MINUS)

9. TCL

- COMMIT: 작업 완료
- ROLLBACK: 철회
- SAVEPOINT identifier : 저장점 생성
- ROLLBACK TO SAVEPOINT identifier: 저장점으로 되돌리기
- InnoDB 은 SAVEPOINT 와 ROLLBACK TO SAVEPOINT 문을 지원
- MySQL 5.1까지의 기본 엔진으로 트랜잭션을 지원하지 않습니다.
- 테이블이 사용하고 있는 엔진 확인

select engine from information_schema.TABLES where table_name='테이블이름' AND table_schema= ' 데이터베이스이름';

9. TCL

- SAVEPOINT 문은 identifier의 이름으로 명명된 트랜잭션 savepoint를 설정
- 현재 트랜잭션이 같은 이름으로 savepoint를 가지고 있다면 예전 savepoint는 삭제되고 새로운 것으로 대체
- ROLLBACK TO SAVEPOINT 문은 명명된 savepoint로 롤백(roll back)
- 현재 트랜잭션이 savepoint후에 레코드들에 대한 변경은 롤백에서 수행되지 않지만, InnoDB는 savepoint 뒤에 메모리에 저장된 열 잠금을 풀지 않으며 명명된 savepoint보다 뒤에 설정된 Savepoints는 삭제
- 만일 ROLLBACK TO SAVEPOINT 문이 다음 에러를 리턴 하면 지정된 이름을 가진 savepoint가 존재 하지 않음을 의미

ERROR 1181: Got error 153 during ROLLBACK

- RELEASE SAVEPOINT 문은 현재 트랜잭션의 savepoint의 집합으로부터 명명된 savepoint를 삭제하며 commit이나 rollback이 발생하지 않으며 Savepoint가 존재 하지 않으면 에러
- savepoint를 지정하지 않는 COMMIT이나 ROLLBACK을 실행하면 현재 트랜잭션의 모든 savepoint는 삭제

10. INDEX

- 데이터를 빠르게 조회하기 위해 사용하는 객체
- primary key 와 unique는 자동 생성
- 생성

create Index 인덱스이름 on 테이블이름(컬럼이름);

- 삭제

drop index 인덱스이름;

- 검색이 인덱스를 사용하는지 확인을 할 때는 앞에 explain을 추가

10. INDEX

```
explain select *
from usertbl
where userid = 'kty';

explain select *
from usertbl
where addr = '서울';

create index idx_addr
on usertbl(addr);

explain select *
from usertbl
where addr = '서울';

drop index idx_addr on usertbl;
```

11. View

- 논리적 가상 테이블로 자주 사용하는 select 구문을 테이블과 같은 하나의 이름으로 만들어 둔 것
- 사용자에게 필요한 데이터만 만들어서 사용하도록 해서 보안을 향상시킬 수 있고 컴파일되서 메인 메모리에 상주한 상태에서 사용하기 때문에 쿼리 실행 속도를 향상 시킬 수 있음
- 생성

  create [or replace] view 뷰이름

  AS select 구문
- 삭제

  drop View 뷰이름

- buytbl 테이블에서 groupname 별로 amount의 합계를 조회하는 구문을 GroupView로 생성

12. PROCEDURE

❖ 프로시저 생성

```
DELIMITER //
create PROCEDURE myproc(vuserid char(15), vname varchar(20)CHARACTER set utf8,
vbirthyear int(11),
vaddr char(100)CHARACTER set utf8, vmobile char(11), vmdate date)
begin
        INSERT INTO usertbl
        VALUES(vuserid, vname, vbirthyear, vaddr, vmobile, vmdate);
end//
DELIMITER ;
```

❖ 프로시저 실행

```
call myproc('BoA', '권보아', 1986, '남양주', '01012341234', '1986-11-5');
```

❖ 확인

```
select * from usertbl;
```