MySQL 강의자료

2023.07

1. MySQL 설치

■ MySQL 설치 (https://dev.mysql.com/downloads/)

MySQL Community version (MySQL Installer for Windows 8.0.33)
중간에 Execute, Next

■ Heidisql 설치 (https://www.heidisql.com/download.php)

Installer, 32/64 bit combined
신규, Root 폴더에 세션 생성
도구>환경설정: 기본, SQL, 격자서식설정 메뉴탭에서 폰트와 글자크기 변경

■ 사용자 생성 및 권한 부여

```
create user 'userID'@'%' identified by 'userpassword';

create user 'ysuser'@'%' identified by 'yspass';
grant all privileges on *.* to 'ysuser'@'%';
flush privileges;
alter user 'ysuser'@'%' identified with mysql_native_password by 'yspass';
```

1. MySQL 설치 – 한글 사용(5.7 이하 버전)

■ 한글 사용(5.7 이하 버전에서는 필요함)

C:\ProgramData\MySQL\MySQL Server 5.7\my.ini 맨 마지막에 추가

```
[client]
default-character-set=utf8

[mysql]
default-character-set=utf8

[mysqld]
character-set-client-handshake = FALSE
init_connect="SET collation_connection = utf8_general_ci"
init_connect="SET NAMES utf8"
character-set-server = utf8

[mysqldump]
default-character-set = utf8
```

■ 한글 확인

```
show variables like 'char%';
ALTER DATABASE [DB명] DEFAULT CHARACTER SET utf8
alter database mysql default character set utf8;
```

■ 사용 가능한 데이터 타입

- Numerical
- Date and Time
- String
- Spatial
- JSON

■ Numerical Data Type

- 정수

타입	저장공간	최소값 (Signed)	최소값 (Unsigned)	최대값 (Signed)	최대값 (Unsigned)
TINYINT	1	-128	0	127	255
SMALLINT	2	-32768	0	32767	65535
MEDIUMINT	3	-2^{23}	0	$2^{23} - 1$	$2^{24} - 1$
INT	4	-2^{31}	0	$2^{31} - 1$	$2^{32} - 1$
BIGINT	8	-2^{63}	0	$2^{63} - 1$	$2^{64} - 1$

- 실수(정확한 값): NUMERIC, DECIMAL

```
salary DECIMAL(5, 2)
```

- 실수(근사치): FLOAT, DOUBLE

■ Date and Time Data Type

- DATE type: YYYY-MM-DD
- DATETIME type: YYYY-MM-DD hh:mm:ss

```
1000-01-01 00:00:00 ~ 9999-12-31 23:59:59
```

- TIMESTAMP type

```
1970-01-01 00:00:01 UTC ~ 2038-01-19 03:14:07 UTC
```

- Time Zone

```
2020-01-01 10:10:10-08:00
```

- Default value

```
dt DATETIME DEFAULT CURRENT_TIMESTAMP
```

■ String Data Type

- CHAR and VARCHAR

Value	CHAR(4)	저장 공간	VARCHAR(4)	저장 공간
''	' '	4	''	1
‘ab’	‘ab ‘	4	‘ab’	3
‘abcd’	‘abcd’	4	‘abcd’	5
‘abcdefgh’	‘abcd’	4	‘abcd’	5

- BLOB(Binary Large OBject)

3. 데이터 조작(SELECT)

```
SELECT select_expr
    [FROM table_references]
    [WHERE where_condition]
    [GROUP BY {col_name | expr | position}]
    [HAVING where_condition]
    [ORDER BY {col_name | expr | position}]
```

■ 데이터 조회 - 조건

```
use world;
show tables;
desc city;

select * from city;

select * from city where countrycode='kor';
select * from city where district='kyonggi';
select name, population from city where district='kyonggi';
select * from city where district='kyonggi' and population>500000;
select name, population from city
    where district='kyonggi' and population>500000;
select district from city where countrycode='kor';
select distinct district from city where countrycode='kor';
select * from mysql.address_book;
select * from city where district='chollanam' or
    district='chollabuk' or district='kwangju';
```

■ 데이터 조회 – 조건 및 순서

```
select * from city where
    countrycode='kor' and population>1000000 and population%2=0;
select * from city where
    countrycode='kor' and population between 1000000 and 2000000;
select * from city where countrycode='kor‘ and name like 'tae%';

select * from city where district='kyonggi' order by name;
select * from city where district='kyonggi' order by name desc;
select * from city where district='kyonggi' order by population desc;
select * from city where countrycode='kor' order by district;
select * from city
    where countrycode='kor' order by district, population;
select * from city
    where countrycode='kor' order by district desc, population;
select * from city
    where countrycode='kor' order by district desc, population desc;
```

■ 데이터 조회 – 함수 이용, 그룹핑

```
select count(*) from city where countrycode='kor';
select sum(population) from city where countrycode='kor';
select avg(population) from city where countrycode='kor';
select max(population) from city;
select min(population) from city;
select min(population), max(population), avg(population),
    sum(population) from city where countrycode = 'kor';

select group_concat(name) from city where district='chungchongnam';
select group_concat(distinct district) from city
    where countrycode='kor';
select district, count(*) from city where countrycode='kor';
select * from city where countrycode='kor' group by district;
select district, count(*) from city
    where countrycode='kor' group by district;
select district, count(*) from city where countrycode='kor'
    group by district having count(*)=6;
select district, count(*) from city where countrycode='kor'
    group by district having count(*)>=6 order by count(*) desc;
```

■ 데이터 조회 – 그룹핑, 조인 예

```
select countrycode, count(*) from city
    group by countrycode having count(*) >=50;
select countrycode, count(*) from city group by countrycode
    having count(*) >=50 order by count(*) desc;

desc country;
desc countrylanguage;

select city.Name, city.Population, country.Name from city
    inner join country on city.CountryCode = country.Code;
select city.Name, city.Population, country.Name from city
    inner join country on city.CountryCode = country.Code
    where city.CountryCode='kor';
select city.Name, city.Population, country.Name from city
    inner join country on city.CountryCode = country.Code
    where city.Population > 7000000;
```

3. 데이터 조작(UPDATE)

■ 데이터 갱신

```
UPDATE 테이블이름
   SET 열1=값1, 열2=값2 …
   WHERE 조건 ;
```

```
use world;
update city set name='Siheung'
    where countrycode='kor' and name='Shihung';
update city set name='Siheung', population=153443
    where countrycode='kor' and name='Shihung';
```

■ 데이터 삽입

```
INSERT [INTO] 테이블[(열1, 열2, …)] VALUES (값1, 값2 …)
```

```
insert into city values (default, 'Gimpo', 'KOR', 'Kyonggi', 200001);
insert into city (name, countrycode, district, population)
    values ('Hwasong', 'KOR', 'Kyonggi', 312345);

insert into city (name, countrycode, district, population)
    values ('Osan', 'KOR', 'Kyonggi', 201234),
    ('Pochon', 'KOR', 'Kyonggi', 156789);
select * from city order by id desc limit 3;
```

■ 대량 데이터 삽입

형식:

```
INSERT INTO 테이블이름 (열 이름1, 열 이름2, …)
   SELECT문  ;
```

```
create table citycopy like city;
show tables;
select * from citycopy;
insert into citycopy select * from city;
select * from citycopy;
```

3. 데이터 조작(DELETE)

■ 데이터 삭제

```
DELETE FROM 테이블이름 WHERE 조건;
```

```
delete from city where id=4082;
select * from city order by id desc limit 3;
```

■ 뷰 생성

```
create view LargeCity as select * from city
    where population>7000000 with check option;
select * from largecity;
show tables;
```

■ Sub Query

```
create view KoreanCity as select id, name, district, population
    from city where countrycode='kor';
select * from KoreanCity ;
select district, name, population from koreancity as c1
    where population > (select avg(population) from koreancity as c2
    where c1.district = c2.district group by district);
```

■ Join

▲ Inner Join

```
select countrylanguage.`*`, country.name from countrylanguage
  inner join country on countrylanguage.countrycode=country.Code
  where language='korean';
```

▲ Outer Join

```
select city.`*`, country.Name from city left outer join country
  on city.CountryCode=country.Code where city.CountryCode='kor';
```

3. 데이터 조작(날짜 형식)

```
CREATE TABLE date_table (
    id int auto_increment primary key,
    dt datetime default current_timestamp
);

INSERT INTO date_table (dt) VALUES
    ('2017-08-28 17:22:21'), ('2017-02-15 10:22:24'),
    ('2017-12-09 22:13:24'), ('2018-07-06 20:15:18');
INSERT INTO date_table VALUES (default, default);

SELECT dt FROM date_table where id=101;
SELECT date_format(dt, '%Y-%m-%d') FROM date_table where id=101;
SELECT date_format(dt, '%h:%i:%s %p')     # %r
    FROM date_table where id=1;

SELECT now(), curdate(), curtime();
SELECT date_add(now(), interval 2 MONTH);
SELECT date_sub(now(), interval 5 DAY);
SELECT to_days('2021-11-18') - to_days(now());  /* from AD */
SELECT dayofweek(datetime) FROM date_table;     /* 일요일: 1 */
```

■ 테이블 생성

```
use test;
create table if not exists address_book (
    no int primary key auto_increment,
    name varchar(10) not null,
    tel varchar(14),
    nickname varchar(20) default '별명',
) auto_increment=10001;

create Table: CREATE TABLE `wp_options` (
    `option_id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
    `option_name` varchar(64) NOT NULL DEFAULT '',
    `option_value` longtext NOT NULL,
    `autoload` varchar(20) NOT NULL DEFAULT 'yes',
    PRIMARY KEY (`option_id`),
    UNIQUE KEY `option_name` (`option_name`)
) ENGINE=MyISAM AUTO_INCREMENT=1203 DEFAULT CHARSET=utf8

show table status;
```

4. 테이블 조작

■ 테이블 조회

```
show tables;
desc address_book;
```

■ 테이블 제거

```
drop table [테이블 명];
create table tmp (
    id int,
    name varchar(10)
);
drop table tmp;
```

■ 테이블 이름 변경

```
rename table [테이블 명] to [새 테이블명];
rename table tmp to tmp_table;
```

■ 테이블 변경

alter table

ADD	컬럼 추가
DROP	컬럼 삭제
CHANGE	컬럼명 변경, 컬럼 자료형 변경
MODIFY	컬럼 순서 바꾸기

▲ 컬럼 추가

```
alter table [테이블 명] add [컬럼명] 자료형;    # 맨 뒤에 추가
alter table address_table add gender char(2) not null;  # 남/여

alter table [테이블 명] add [컬럼명] 자료형 first; # 맨 앞에 추가
alter table [테이블 명] add [컬럼명] 자료형 after [앞 컬럼명];
/* 지정 컬럼 뒤에 추가 */
```

▲ 컬럼 삭제

```
alter table [테이블 명] drop [컬럼명];
```

■ 테이블 변경

alter table

ADD	컬럼 추가
DROP	컬럼 삭제
CHANGE	컬럼명 변경, 컬럼 자료형 변경
MODIFY	컬럼 순서 바꾸기

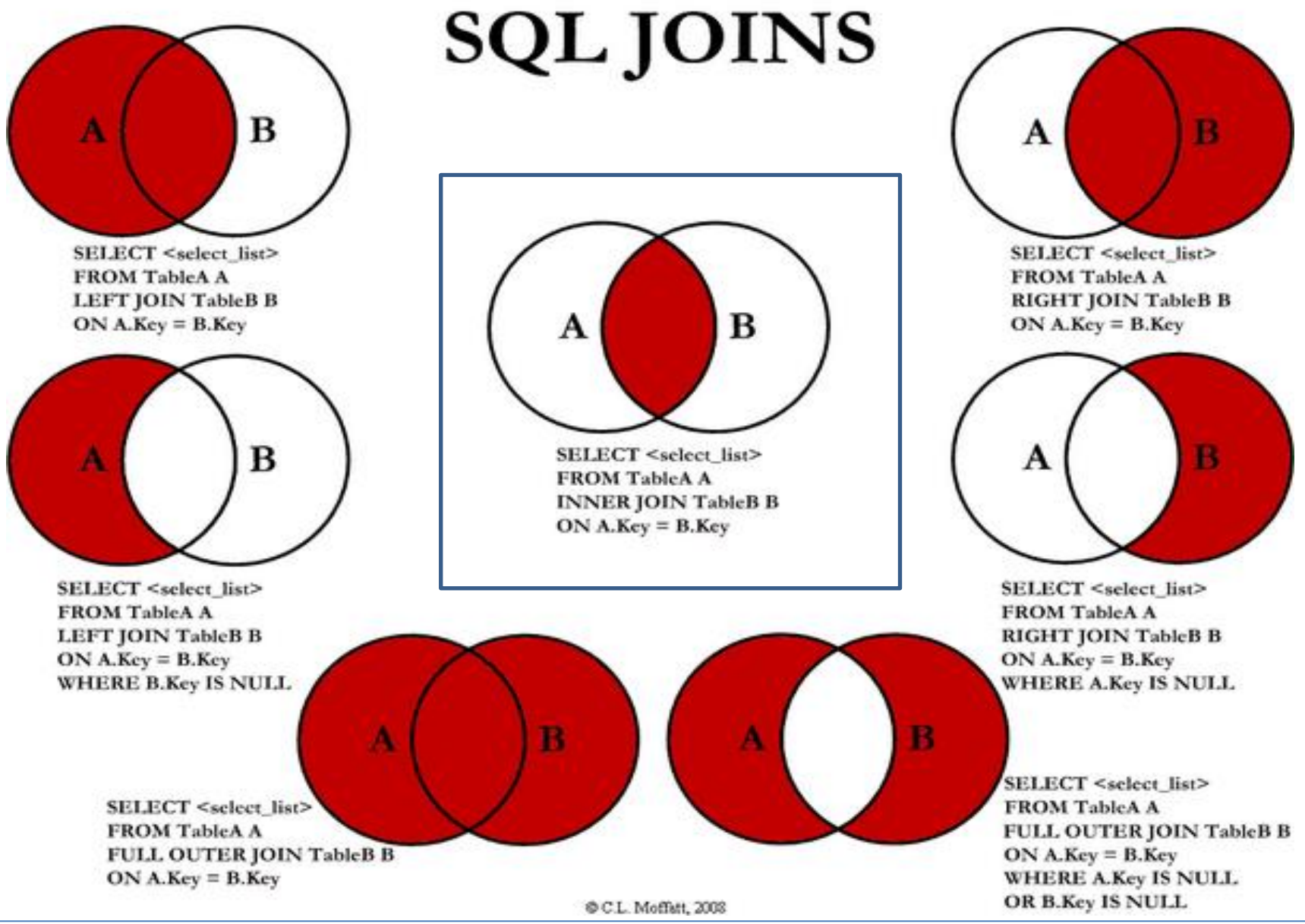
▲ 컬럼명 변경, 컬럼 자료형 변경

```
alter table [테이블 명] change [기존 컬럼명] [새 컬럼명] 자료형;
alter table [테이블 명] change [컬럼명] [컬럼명] 새 자료형;
alter table address_book change no aid int(8);
alter table address_book
                change aid aid int(4) unsigned auto_increment;
```

▲ 컬럼 순서 바꾸기

```
alter table [테이블 명] modify [컬럼명] 자료형 first;
alter table [테이블 명] modify [컬럼명] 자료형 after [다른 컬럼명];
alter table address_book modify gender char(2) not null after name;
```

5. 테이블 조인(Join)


SQL JOINS
A
B
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
© C.L. Moffatt, 2008

■ 테이블 생성

```
CREATE TABLE song (
    sid INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(32) NOT NULL,
    lyrics VARCHAR(32)
) AUTO_INCREMENT=101;

CREATE TABLE girl_group (
    gid INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(32) NOT NULL,
    debut DATE NOT NULL,
    hit_song_id INT                    # foreign key(외래키)
) AUTO_INCREMENT=1001;
```

■ 데이터 삽입

```
INSERT INTO song (title, lyrics)
    VALUES ('Tell Me', 'tell me tell me tetetete tel me'),
    ('Gee', 'GEE GEE GEE GEE GEE BABY BABY'),
    ('미스터', '이름이 뭐야 미스터'),
    ('Abracadabra', '이러다 미쳐 내가 여리여리'),
    ('8282', 'Give me a call Baby baby'), ('기대해', '기대해'),
    ('I Don\'t care', '다른 여자들의 다리를'),
    ('Bad Girl Good Girl', '앞에선 한 마디 말도'), ('피노키오', '뉴예삐오'),
    ('별빛달빛', '너는 내 별빛 내 마음의 별빛'),
    ('A', 'A 워오우 워오우워 우우우'),
    ('나혼자', '나 혼자 밥을 먹고 나 혼자 영화 보고'), ('LUV', '설레이나요 '),
    ('짧은치마', '짧은 치마를 입고 내가 길을 걸으면'),
    ('위아래', '위 아래 위위 아래'), ('Dumb Dumb', '너 땜에 하루종일');

INSERT INTO girl_group (name, debut)
    VALUES ('원더걸스', '2007-02-10', 101),
    ('소녀시대', '2007-08-02', 102), ('카라', '2009-07-30', 103),
    ('브라운아이드걸스', '2008-01-17', 104), ('다비치', '2009-02-27', 105),
    ('2NE1', '2009-07-08', 106), ('f(x)', '2011-04-20', 108),
    ('시크릿', '2011-01-06', 109), ('레인보우', '2010-08-12', 110),
    ('애프터 스쿨', '2009-11-25', 120), ('포미닛', '2009-08-28', 121);
```

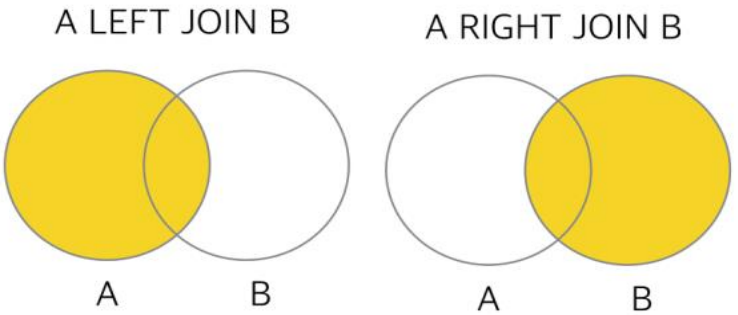
5. 테이블 조인(Join)

■ Inner Join

```
SELECT gg.gid, gg.name, s.title
     FROM girl_group AS gg
     INNER JOIN song AS s                    # JOIN song AS s
     ON s.sid = gg.hit_song_id;
```

■ Left Outer, Right Outer Join

```
SELECT gg.gid, gg.name, s.title
     FROM girl_group AS gg
     LEFT OUTER JOIN song AS s               # LEFT JOIN song AS s
     ON s.sid = gg.hit_song_id;

SELECT s.sid, s.title, gg.name
     FROM girl_group AS gg
     RIGHT OUTER JOIN song AS s              # RIGHT JOIN song AS s
     ON s.sid = gg.hit_song_id;
```



■ 연습 문제

1) 2009년도에 데뷔한 걸그룹 정보를 조회
   (where debut between '2009-01-01' and '2009-12-31' 이용)

2) 2009년도에 데뷔한 걸그룹의 히트송은?
   (걸그룹 이름, 데뷔일, 히트송)

3) 대륙별로 국가숫자, GNP의 합, 평균 국가별 GNP는?

4) 아시아 대륙에서 인구가 가장 많은 도시 10개를 내림차순으로 보여줄 것
   (대륙명, 국가명, 도시명, 인구수)

5) 전 세계에서 인구가 가장 많은 10개 도시에서 사용하는 공식언어는?
   (도시명, 인구수, 언어명)

■ 테이블 Export/Import (in HeidiSQL)

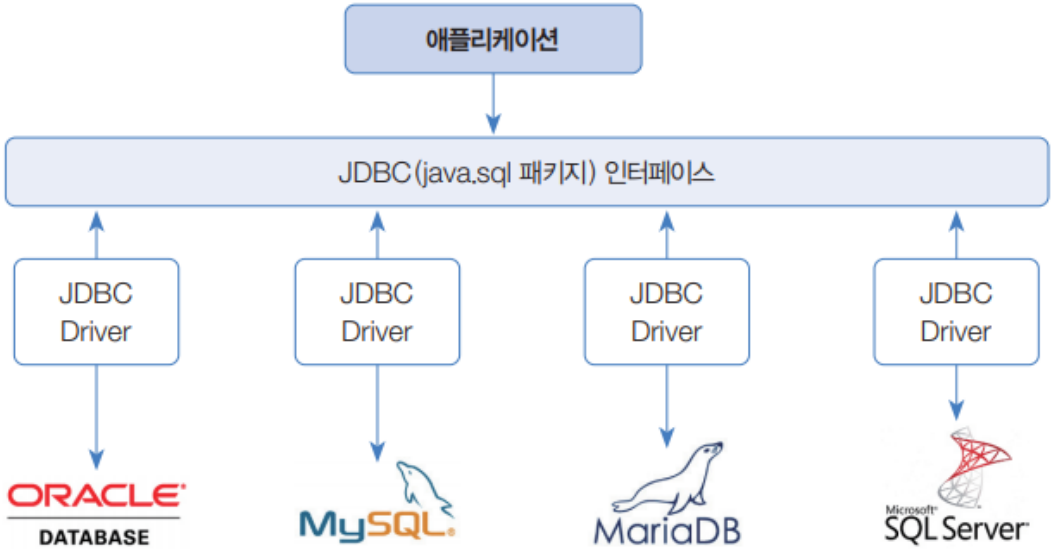
```
SHOW VARIABLES LIKE "secure_file_priv";
    ➔ C:/ProgramData/MySQL/MySQL Server 8.0/Uploads

SELECT * FROM song INTO OUTFILE
    'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/song.csv'
    FIELDS TERMINATED BY ','
    ENCLOSED BY '"'
    LINES TERMINATED BY '\n';

TRUNCATE song;

LOAD DATA INFILE
    'c:/ProgramData/MySQL/MySQL Server 8.0/Uploads/song.csv'
    INTO TABLE song
    FIELDS TERMINATED BY ','
    ENCLOSED BY '"'
    LINES TERMINATED BY '\n';

SELECT * FROM song;
```

■ Key의 종류

- 후보키 (Candidate Key) :
  - 테이블을 구성하는 열 중에서 유일하게 식별할 수 있는 열
- 기본키 (Primary Key) :
  - 테이블에서 유일하게 식별하기 위해 사용하는 키
- 대체키 (Alternate Key) :
  - 후보키 중 기본키를 제외한 나머지 후보키
- 외래키 (Foreign Key) :
  - 테이블 내의 열 중 다른 테이블의 기본키를 참조하는 열
- 슈퍼키 (Super Key) :
  - 슈퍼키 또는 합성키라 불림
  - 2개 이상의 열이 합쳐서 기본키로 사용하는 것

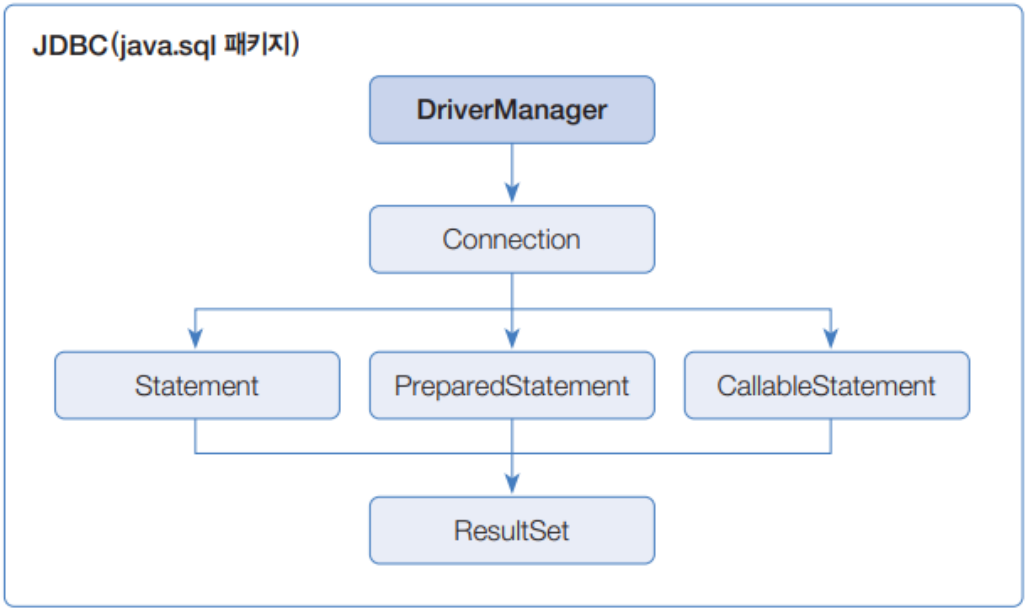
■ JDBC 라이브러리

- 자바는 데이터베이스(DB)와 연결해서 데이터 입출력 작업을 할 수 있도록 JDBC 라이브러리 (java.sql 패키지)를 제공
- JDBC는 데이터베이스 관리시스템(DBMS)의 종류와 상관없이 동일하게 사용할 수 있는 클래스와 인터페이스로 구성



■ JDBC Driver

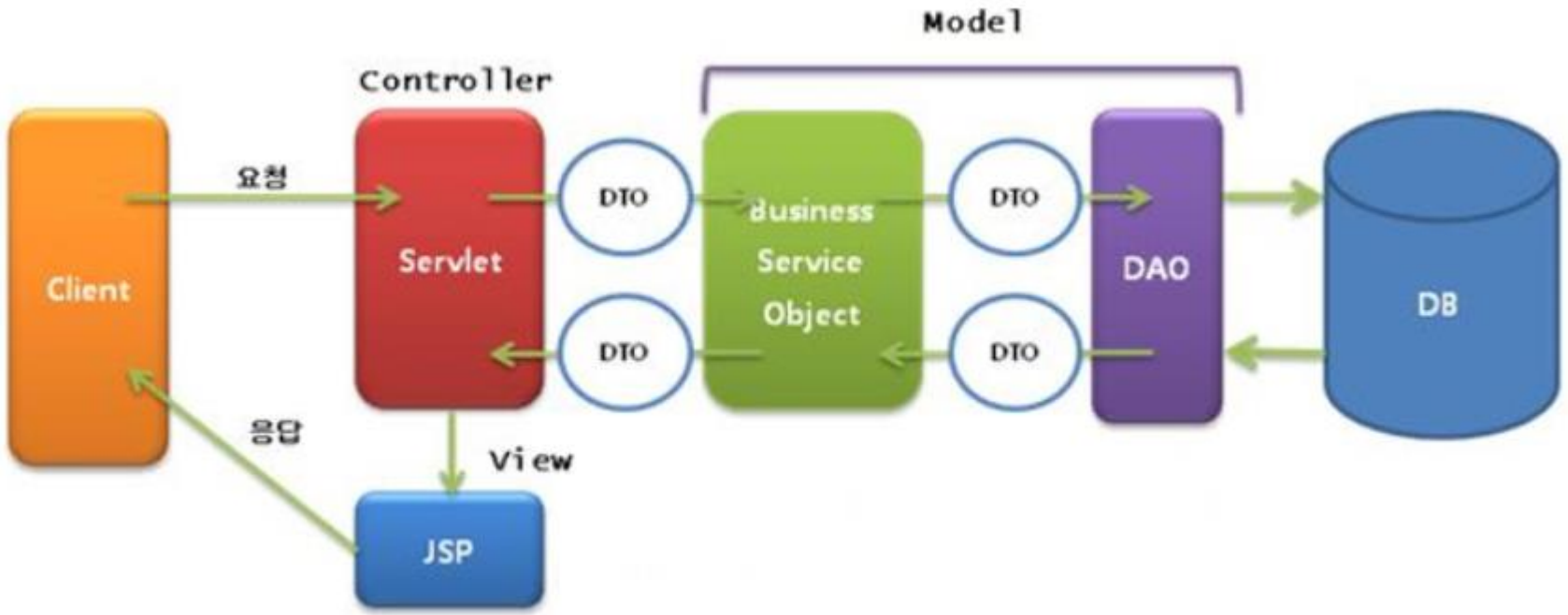
- JDBC 인터페이스를 구현한 것으로, DBMS마다 별도로 다운로드받아 사용
  (예: `mysql-connector-java-8.0.33.jar`)
- DriverManager 클래스: JDBC Driver를 관리하며 DB와 연결해서 Connection 구현 객체를 생성
- Connection 인터페이스: Statement, PreparedStatement, CallableStatement 구현 객체를 생성하며, 트랜잭션 처리 및 DB 연결을 끊을 때 사용
- Statement 인터페이스:
  SQL의 DDL과 DML 실행 시 사용
- PreparedStatement:
  SQL의 DDL, DML 문 실행 시 사용.
  매개변수화된 SQL 문을 사용하여
  편리성과 보안성 유리
- CallableStatement: DB에 저장된
  프로시저와 함수를 호출
- ResultSet: DB에서 가져온 데이터를 읽음



■ 샘플 프로그램

```
Connection conn = null;
Statement st = null;
try {
    conn = DriverManager.getConnection(
                    "jdbc:mysql://localhost:3306/test",
                    "ysuser", "yspass");
    st = conn.createStatement();
    String sql = "select * from dept;";
    ResultSet rs = st.executeQuery(sql);
    while (rs.next()) {
        int deptId = rs.getInt(1);
        String name = rs.getString(2);
        System.out.printf("%d %s\n", deptId, name);
    }
    rs.close();
} catch (SQLException e) {
    e.printStackTrace();
} finally {
    try {
        st.close();
        conn.close();
    } catch (SQLException se) {
        se.printStackTrace();
    }
}
```

■ MVC(Model-View-Controller)



■ DAO(Data Access Object)

```
List<UserDTO> getUserList(int offset)

UserDTO getUserInfo(String uid)

void insertUser(UserDTO user)

void updateUser(UserDTO user)

void deleteUser(String uid)
```

■ DTO(Data Transfer Object)

```
import java.time.LocalDate;

public class UserDTO {
    private String uid;
    private String password;
    private String name;
    private String email;
    private LocalDate regDate;
    private int isDeleted;

    public UserDTO() {}
    public UserDTO(String uid, String password,
            String name, String email,
            LocalDate regDate, int isDeleted) {
        this.uid = uid;
        this.password = password;
        this.name = name;
        this.email = email;
        this.regDate = regDate;
        this.isDeleted = isDeleted;
    }
    ...
}
```

8. NodeJS 에서 MySQL 사용법

- **프로그램 설치**

```
npm install mysql
```

- **샘플 프로그램**

```
const mysql = require('mysql');
let connection = mysql.createConnection({
    host     : 'localhost',
    user     : 'ysuser',
    password : 'yspass',
    database : 'world'
});

connection.connect();

let sql = 'SELECT * FROM city WHERE population > 9000000;';
connection.query(sql, function (error, results, fields) {
    if (error)
        throw error;
    console.log(results);
});

connection.end();
```

- **Config file(mysql.json)을 이용한 샘플 프로그램**

```
const fs = require('fs');
const mysql = require('mysql');
const config = fs.readFileSync('./mysql.json');

const connection = mysql.createConnection(config);
connection.connect();

let sql = 'SELECT * FROM city WHERE population > 9000000;';
connection.query(sql, function (error, results, fields) {
    if (error) throw error;
    for (let city of results)
        console.log(city.Name, city.CountryCode, city.Population);
});

connection.end();
```

- **Insert 샘플 프로그램**

```
connection.connect();

let player = "최형우";
let backNo = "34";
let position = "외야수";
let params = [player, backNo, position];

const sql = 'insert into customer(player, backNo, position) \
             values (?, ?, ?, ?, ?)';
connection.query(sql, params, function (error, results) {
    if (error)
        throw error;
});

connection.end();
```

8. NodeJS 에서 MySQL 사용법

```
getDB(params, callback) {
    connection.connect();
    let sql = 'SELECT * FROM ...;';
    connection.query(sql, params, function (error, results, fields) {
        if (error)
           throw error;
        callback(results);
    });
    connection.end();
}

writeDB(params, callback) {
    connection.connect();
    let sql = 'INSERT INTO ...;';
    connection.query(sql, params, function (error, fields) {
        if (error)
           throw error;
        callback();
    });
    connection.end();
}
```

9. Python 에서 MySQL 사용법

■ 프로그램 설치

```
conda install pymysql
```

■ 샘플 접속 프로그램

```
import json
with open('mysql.json', 'r') as file:
    config_str = file.read()
config = json.loads(config_str)

import pymysql
conn = pymysql.connect(
    host = config['host'],
    user = config['user'],
    password = config['password'],
    database = config['database'],
    port = config['port']
)
```

mysql.json

```
{
    "host": "localhost",
    "user": "ysuser",
    "password": "yspass",
    "database": "world",
    "port": 3306
}
```

9. Python 에서 MySQL 사용법

■ 테이블 생성

```
cur = conn.cursor()
sql_user = '''
    create table if not exists users (
        uid varchar(20) not null primary key,
        pwd char(44) not null,
        uname varchar(20) not null,
        reg_date datetime default current_timestamp,
        is_deleted int default 0
    );
'''
cur.execute(sql_user)
```

■ 테이블 구조/이름 변경

```
cur.execute('ALTER TABLE users ADD COLUMN email varchar(40)')
```

■ 테이블 삭제

```
cur.execute('DROP TABLE users')
```

■ 데이터 삽입

```
# 기본 스트링 쿼리
cur = conn.cursor()
cur.execute("INSERT INTO users(uid, uname) VALUES('admin', '관리자');")
cur.execute("INSERT INTO users(uid, uname) VALUES \
                ('eskim', '김은숙'), ('wjlee', '이우정');")
conn.commit()

# 파라메터: 튜플 사용
uid = 'djy'
uname = '대조영'
cur = conn.cursor()
sql = 'INSERT INTO users(uid, uname) VALUES (%s, %s);'
cur.execute(sql, (uid, uname))

# 튜플 리스트 사용
users = (('gdhong', '홍길동'), ('jbpark', '박재범'))
cur = conn.cursor()
sql = 'INSERT INTO users(uid, uname) VALUES (%s, %s);'
cur.executemany(sql, users)
```

■ 데이터 조회

```
sql = """SELECT uid, uname, DATE_FORMAT(reg_date, '%Y-%m-%d') AS reg_date
            FROM users WHERE isDeleted=0;"""
# 순회 조회
cur = conn.cursor()
cur.execute(sql)
for row in cur:
    print(row)

# 단건 조회
cur = conn.cursor()
cur.execute(sql)
row = cur.fetchone()

# 다건 조회
rows = cur.fetchmany(2)

# 모두 조회
rows = cur.fetchall()
for row in rows:
    print(row)
```

■ 데이터 검색

```
# 기본 스트링 쿼리
cur = conn.cursor()
cur.execute("SELECT uname FROM users WHERE uid='eskim';")
rows = cur.fetchall();
for row in rows:
    print(row)

# Placeholder
cur = conn.cursor()
uid = 'wjlee'
cur.execute('SELECT uname FROM users WHERE uid=%s;', (uid,))
writer = cur.fetchone()
print(writer[0])                    # 이우정
```