

제9장 : 데이터베이스 접속

DB를 사용했을 때의 이점

- 처리 속도의 향상
- 동시에 여러 프로그램에서 접속 가능
- 여러 처리에 강함

- 검색, 삽입, 갱신, 삭제 4가지 기본 동작
- 4가지 동작은 각각 **select, insert, update, delete** 의 SQL (Structured Query Language) 문을 DBMS (DataBase Management System)에 송신하여 처리 함

SQL 학습 : <https://www.w3schools.com/sql/default.asp>

- 일반적으로는 서버, 클라이언트로 네트워크 통신을 사용할 일이 많음
- Oracle, DB2, MySQL, MSSql ...

What is a DBMS?



- Java에서는 JDBC (Java DataBase Connectivity) 라는 전용 DB 조작용 API를 제공함.

자주 사용되는 4가지 클래스

클래스명	용도
DriverManager	DBMS 로 접속 준비
Connection	DBMS 로 접속이나 연결 해제
PreparedStatement	SQL 문을 송신
ResultSet	DBMS 로부터 결과를 받을 때

- JDBC는 여러가지 DBMS에 대한 접속용 드라이버를 제공

1. MySQL 설치 (5.6 버전으로 연습)
 - a. <https://dev.mysql.com/downloads/mysql/5.6.html>
2. MySQL 실행
 - a. `mysql -u root -p`
3. JDBC 드라이버 다운로드
 - a. <https://dev.mysql.com/downloads/connector/j/5.1.html>
4. Java 프로젝트에 bin-jar 파일 의존성 추가 및 예제 코드 실행하여 연결 확인
 - a. <https://whitepaek.tistory.com/18>


MySQL 8.0 이후 발생하는 에러

<https://joont92.github.io/java/java-mysql-%EC%97%B0%EB%8F%99%EC%8B%9C-%EC%98%A4%EB%A5%98/>

1. (윈도우) SQLyog : <https://all-record.tistory.com/94>
2. (맥) Sequel Pro : <https://www.sequelpro.com/>
3. (MySQL 전용) MySql Workbench : <https://www.mysql.com/products/workbench/>

- 서버 : 제공하는 쪽
- 클라이언트 : 제공받는 쪽

1. **test** 이름의 데이터베이스 생성 (검색어 : `mysql sql create database`)
2. 영웅 정보를 저장할 **Hero** 테이블 작성 (검색어 : `mysql create table sql`)
 - a. 쿼리 확인 후 **SQL** 클라이언트 작성하는 것이 정신건강에 유리함
3. **name**, **hp** 속성을 가진 영웅 정보를 여러 개 추가 (검색어 : `mysql insert sql`)

TABLES	
	Hero

id	name	hp
1	홍길동	50
2	한석봉	80

1. DBMS에 송신할 SQL 문을 준비
2. 쿼리를 완성
3. **ResultSet** 객체를 사용하여 결과를 얻음

```
// SQL문 준비
PreparedStatement pstmt = con.prepareStatement("SELECT * FROM Hero WHERE hp > ?");
// 쿼리 완성
pstmt.setInt(1, 50);
// 결과 얻기
ResultSet result = pstmt.executeQuery();

// 결과를 처리

// 닫기
result.close();
pstmt.close();
```

1. ResultSet 객체 조작

메서드	설명
boolean next()	바라보는 행을 이동. 성공시 true, 실패(마지막행)시 false
int getInt(int colIndex) int getInt(String colName)	지정한 컬럼(열)의 정수값을 얻음. 1부터 시작 또는 컬럼명으로도 지정 가능
String getString(int colIndex) String getString(String colName)	지정한 컬럼(열)의 문자열값을 얻음. 1부터 시작 또는 컬럼명으로도 지정 가능

```
// 결과를 처리
while (result.next()) {
    ... System.out.println(result.getString(columnLabel: "name") + "," + result.getString(columnLabel: "hp"));
}
```

INSERT, UPDATE, DELETE 문은 갱신용 SQL문이고 처리 후에 **몇 행이 처리되었나**를 숫자로 반환함

1. 송신할 SQL문 준비
2. SQL 완성
3. 처리결과를 반환

```
// SQL문 준비
PreparedStatement pstmt = con.prepareStatement("DELETE FROM Hero WHERE hp > ? OR name = ?");
// SQL 완성
pstmt.setInt(1, 80);
pstmt.setString(2, "홍길동");
// 결과 반환
int result = pstmt.executeUpdate();
// 결과 처리
if (result > 0) {
    System.out.println(result + " 건의 영웅이 삭제됨");
} else {
    System.out.println("해당하는 영웅이 없음");
}
// 닫기
pstmt.close();
```

1개 이상의 **SQL** 조작을 1개로 그룹화하여 처리하는 것

- 도중에 에러가 나면 모든 동작을 원래대로 되돌림
- 은행 계좌 처리 등
 - A계좌에서 B계좌로 이체시
 - A계좌에서 -10000원 O, B계좌 +10000원 X 인 경우 에러이기 때문에 롤백(rollback) 됨
 - 두 동작이 한 셋트로 성공시만 DB에 업데이트 반영 됨

```
// 연결

try {
    ...// 수동 커밋으로 변경
    con.setAutoCommit(false);

    ...// SQL 송신 처리

    con.commit();...// 커밋
} catch (SQLException e) {
    try {
        con.rollback();...// 롤백
    } catch (SQLException e2) {
        ...// 롤백 실패 처리
        e2.printStackTrace();
    } finally {
        if (con != null) {
            try {
                con.close();
            } catch (SQLException e3) {
                e3.printStackTrace();
            }
        }
    }
}

// 닫기
```

연습문제 9-1

MySQL 데이터베이스에 다음과 같은 ITEM 테이블을 작성하시오.

NAME	PRICE	WEIGHT
약초	5	2
해독제	7	2

이 데이터를 저장하는 Item 클래스를 정의하시오.

다음과 같은 Main 클래스가 있습니다. 이 클래스에서 사용하는 ItemDAO 클래스를 작성하시오.

```
class Main {  
    public static void main(String[] args) {  
        System.out.println("1원 이상의 아이템 목록");  
  
        // SELECT * FROM ITEM WHERE PRICE > 1  
        ArrayList<Item> items = ItemDAO.findByMinimumPrice(1);  
        for(Item item : items) {  
            System.out.println(item.getName() + ", " +  
                item.getPrice() + ", " + item.getWeight());  
        }  
    }  
}
```