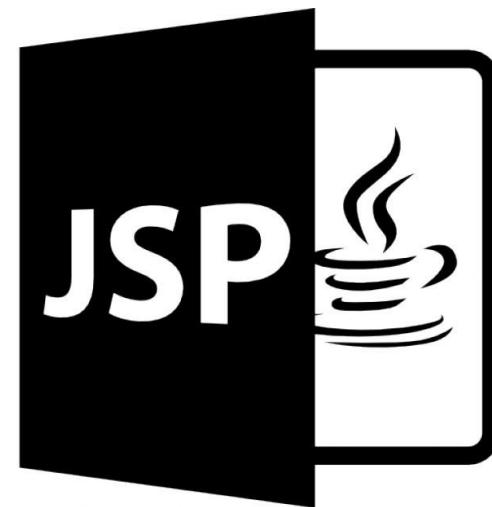


11장. JDBC 연동



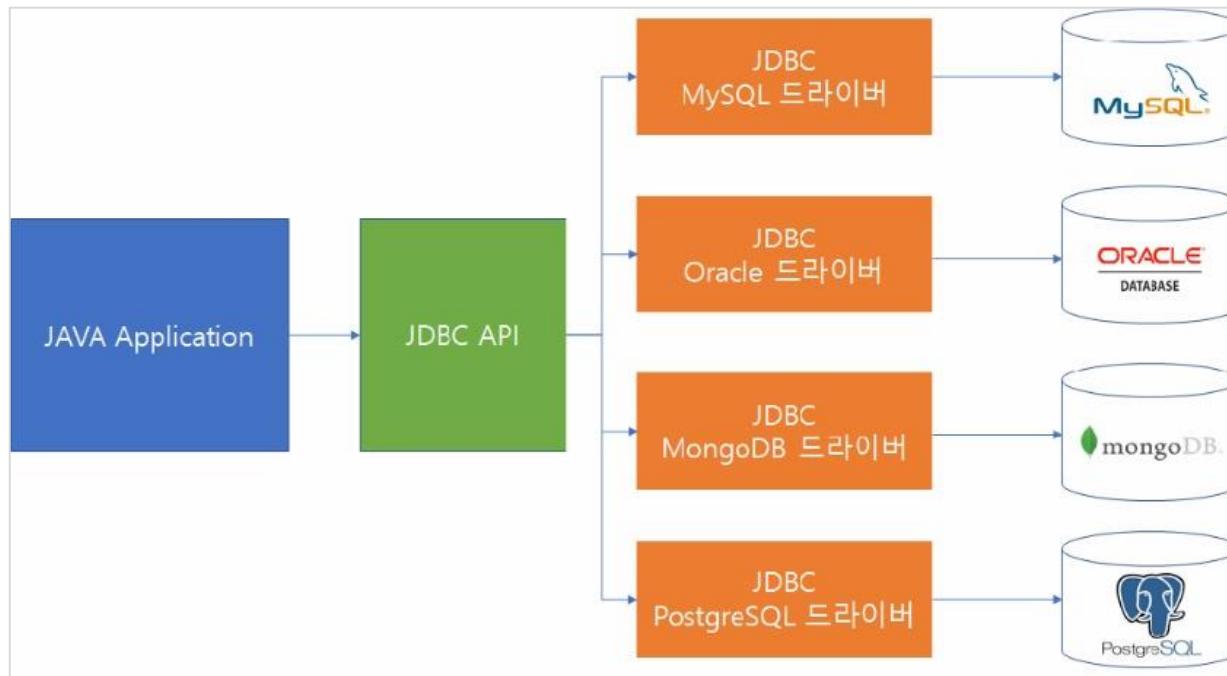
JDBC – oracle, mysql



JDBC(Java Database Connectivity)

◆ JDBC(Java Database Connectivity) 정의와 사용

- 자바 애플리케이션에서 DBMS에 연결해주는 기능을 갖는 API이다.
- 오라클, MySQL, MS-SQL 개발사가 구현한 '드라이버'가 존재함



JDBC(Java Database Connectivity)

◆ JDBC를 이용한 데이터베이스 연동하기

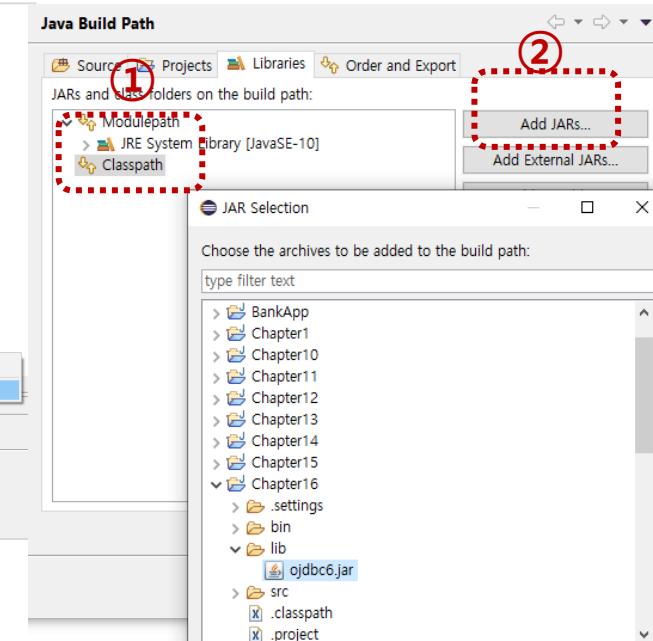
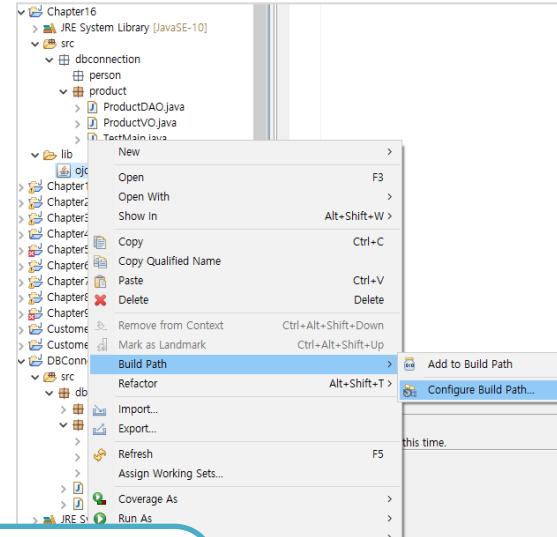
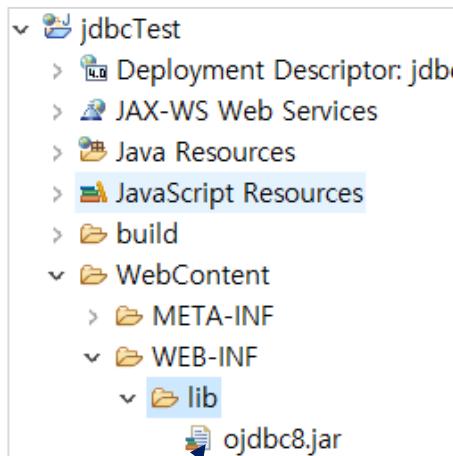
ojdbc 드라이버 구하기 - sql 디벨로퍼 설치 경로



JDBC(Java Database Connectivity)

◆ JDBC를 이용한 데이터베이스 연동하기

Ojdbc.jar 파일을 이클립스 프로젝트에 복사하기

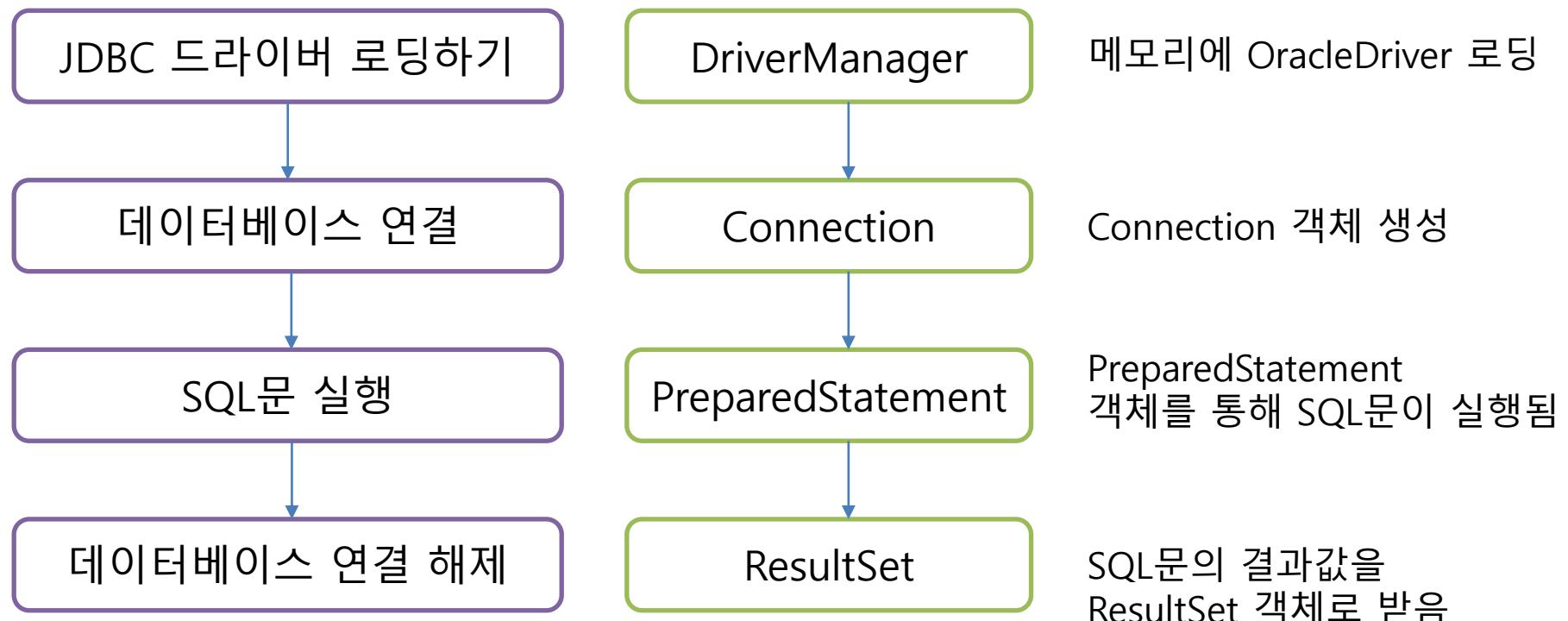


1. 프로젝트의 lib폴더에 .jar파일 복사
2. 클래스 패스 설정

프로젝트 > 우측마우스 > Build Path > Configure Build Path > Libraries(Classpath) > Add JARs

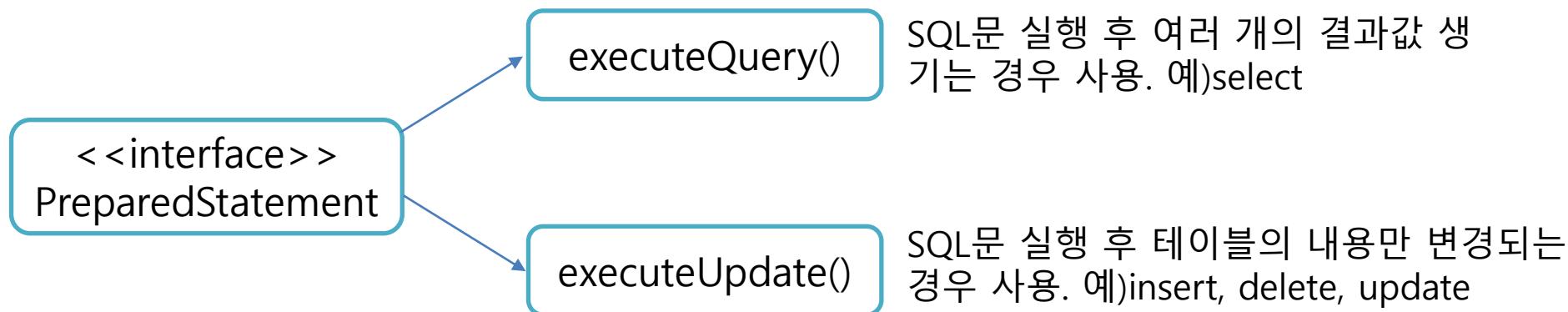
JDBC(Java Database Connectivity)

➤ 데이터베이스 연결 순서

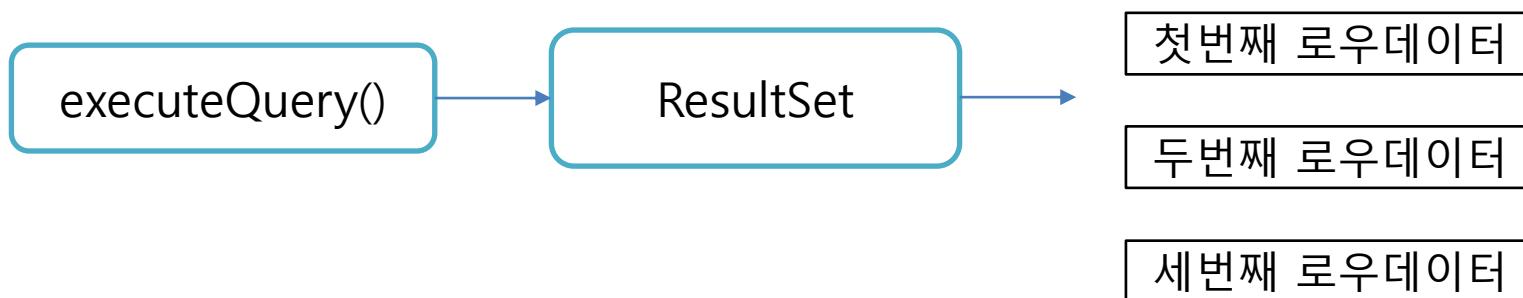


JDBC[Java Database Connectivity]

➤ Statement 객체 살펴보기



executeQuery() 실행 후 반환 되는 레코드셋



DB에서 테이블 생성하기

◆ HRDB에서 t_student 테이블 생성

```
-- t_student 테이블 생성
CREATE TABLE t_student(
    studentId NUMBER(3),
    studentName VARCHAR2(10) NOT NULL,
    PRIMARY KEY(studentId)
);
```

```
-- 학생 추가
INSERT INTO t_student VALUES (101, '이강');
INSERT INTO t_student VALUES (102, '김산');

-- 학생 조회
SELECT *
FROM t_student
ORDER BY studentId;

COMMIT;
```

STUDENTID	STUDENTNAME
101	이강
102	김산



JDBC(Java Database Connectivity)

◆ 애플리케이션에서 DB에 접속 및 데이터 삽입하기

```
package com.jdbcTest;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class DBConnection {
    private static String driverClass = "oracle.jdbc.OracleDriver"; //드라이버 이름
    private static String url = "jdbc:oracle:thin:@localhost:1521:xe"; //DB 주소(경로)
    private static String username = "HR"; //사용자
    private static String password = "1234"; //비밀번호

    private static Connection conn;
    private static PreparedStatement pstmt;
```



JDBC(Java Database Connectivity)

```
public static void main(String[] args) {
    try {
        Class.forName(driverClass);
        conn = DriverManager.getConnection(url, username, password);
        System.out.println("DB 연결 성공!!");

        //자료 삽입
        String sql = "INSERT INTO t_student VALUES (104, '강태양')";
        System.out.println("학생 추가!!!");
        pstmt = conn.prepareStatement(sql); //sql을 처리하는 pstmt 객체 생성
        pstmt.executeUpdate();

    } catch (Exception e) {
        e.printStackTrace();
    }
}

} finally {
    if(pstmt != null) {
        try {
            pstmt.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    if(conn != null) {
        try {
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```



JDBC[Java Database Connectivity]

◆ 애플리케이션에서 DB에 접속하고 조회하기

```
public class DBSelect {  
    private static String driverClass = "oracle.jdbc.OracleDriver";  
    private static String url ="jdbc:oracle:thin:@localhost:1521:xe";  
    private static String username = "HR";  
    private static String password = "1234";  
  
    private static Connection conn;  
    private static PreparedStatement pstmt;  
    private static ResultSet rs;  
  
    public static void main(String[] args) {  
        try {  
            Class.forName(driverClass);  
            conn = DriverManager.getConnection(url, username, password);  
            System.out.println("DB 연결 성공!!");  
  
            //자료 조회  
            String sql = "SELECT * FROM t_student ORDER BY studentId";  
            pstmt = conn.prepareStatement(sql);  
            rs = pstmt.executeQuery();  
            while(rs.next()) {  
                System.out.print("학번 : " + rs.getInt("studentId"));  
                System.out.println(", 이름 : " + rs.getString("studentName"));  
            }  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```



JDBC[Java Database Connectivity]

◆ 애플리케이션에서 DB에 접속하고 조회하기

```
    } finally {
        if(rs != null) {
            try {
                rs.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        if(stmt != null) {
            try {
                stmt.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        if(conn != null) {
            try {
                conn.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

DB 연결 성공!!

학번 : 101, 이름 : 이강
학번 : 102, 이름 : 김산
학번 : 103, 이름 : 산들
학번 : 104, 이름 : 강태양



JDBC 연결 테스트(Connection Test)

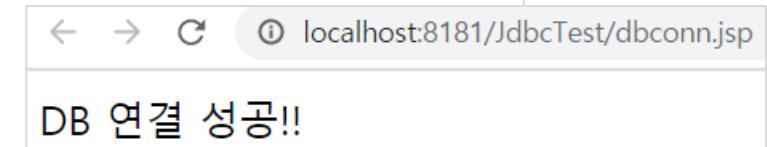
◆ JSP와 DB에 연결하기

```
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.Connection"%>

<%
    String driverClass = "oracle.jdbc.OracleDriver"; //드라이버 이름
    String url = "jdbc:oracle:thin:@localhost:1521:xe"; //DB 주소(경로)
    String username = "HR"; //사용자
    String password = "1234"; //비밀번호

    Connection conn = null;

    try{
        Class.forName(driverClass);
        conn = DriverManager.getConnection(url, username, password);
        out.println("DB 연결 성공!!!");
    }catch(Exception e){
        e.printStackTrace();
    }finally{
        if(conn != null)
            conn.close();
    }
%>
```



JDBC 연결 테스트(Connection Test)

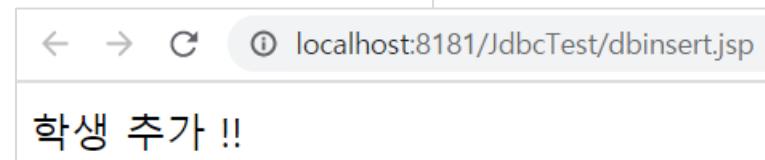
◆ 학생 추가(삽입)하기

```
<%
    String driverClass = "oracle.jdbc.OracleDriver"; //드라이버 이름
    String url ="jdbc:oracle:thin:@localhost:1522:xe"; //DB 주소(경로)
    String username = "HR"; //사용자
    String password = "1234"; //비밀번호

    Connection conn = null;
    PreparedStatement pstmt = null;

    try{
        Class.forName(driverClass);
        conn = DriverManager.getConnection(url, username, password);

        String sql = "INSERT INTO t_student VALUES (201, '박화성')";
        out.println("학생 추가 !!");
        pstmt = conn.prepareStatement(sql);
        pstmt.executeUpdate();
    }catch(Exception e){
        e.printStackTrace();
    }finally{
        if(pstmt != null)
            pstmt.close();
        if(conn != null)
            conn.close();
    }
}
```



JDBC 연결 테스트(Connection Test)

◆ 학생 목록 보기

```
ResultSet rs = null;

try{
    Class.forName(driverClass);
    conn = DriverManager.getConnection(url, username, password);

    String sql = "SELECT * FROM t_student ORDER BY studentId";
    out.println("<h2>학생 명단</h2>");
    pstmt = conn.prepareStatement(sql);
    rs = pstmt.executeQuery();
    out.println("<table border=1 width=200>");
    out.println("<tr><td>학번</td><td>이름</td></tr>");
    while(rs.next()){
        out.println("<tr><td>" + rs.getInt("studentId") + "</td>");
        out.println("    <td>" + rs.getString("studentName") + "</td></tr>");
    }
    out.println("</table>");
} catch(Exception e){
    e.printStackTrace();
} finally{
    if(rs != null)
        rs.close();
    if(pstmt != null)
        pstmt.close();
}
```

학생 명단

학번	이름
101	이강
102	김산
103	산들
104	강태양
201	박화성



MySQL 데이터베이스

웹 애플리케이션에서 데이터베이스와의 연동은 필수적인 작업이다. 이런 상호작용을 위해서는 데이터베이스 관리 시스템이 설치되어야 한다.

▶ MySQL 설치하기

다운로드 하기 : <https://dev.mysql.com/downloads/installer/>

MySQL Community Downloads

- MySQL Yum Repository
- MySQL APT Repository
- MySQL SUSE Repository
- MySQL Community Server
- MySQL Cluster
- MySQL Router
- MySQL Shell
- MySQL Workbench
- MySQL Installer for Windows
- MySQL for Visual Studio

- C API (libmysqlclient)
- Connector/C++
- Connector/J

Recommended Download:

MySQL Installer
for Windows

All MySQL Products. For All Windows Platforms.
In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI

[Go to Download Page >](#)



MySQL 데이터 베이스

▷ MySQL 설치하기

다운로드 하기 : <https://dev.mysql.com/downloads/installer/>

버전 – MySQL 5.7.21

The screenshot shows two side-by-side web pages. The left page is titled 'MySQL Product Archives' and displays a warning about old versions. It has dropdown menus for 'Product Version' (set to 5.7.21) and 'Operating System' (set to Microsoft Windows). The right page is titled 'MySQL Community Downloads' and shows the 'MySQL Installer' section. A red arrow points from the 'Product Version' dropdown on the left to the 'General Availability (GA) Releases' tab on the right, which is highlighted with a red circle labeled '1'. Below this tab, the 'MySQL Installer 8.0.22' section is visible, along with a dropdown menu for 'Select Operating System' set to 'Microsoft Windows'. At the bottom of the right page, there are two download links for 'Windows (x86, 32-bit), MSI Installer' files, both dated Jan 10, 2018.

File Type	Date	Size	Action
Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-5.7.21.0.msi)	Jan 10, 2018	18.6M	Download
Windows (x86, 32-bit), MSI Installer (mysql-installer-community-5.7.21.0.msi)	Jan 10, 2018	370.8M	Download

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

MySQL open source software is provided under the GPL License.



MySQL 데이터 베이스

▷ MySQL 설치하기

Please select the Setup Type that suits your use case.

- Developer Default
Installs all products needed for MySQL development purposes.
- Server only
Installs only the MySQL Server product.
- Client only
Installs only the MySQL Client products, without a server.
- Full
Installs all included MySQL products and features.
- Custom
Manually select the products that should be installed on the system.

Setup Type Description

Allows you to select exactly which products you would like to install. This also allows to pick other server versions and architectures (depending on your OS).

Please select the products and features you would like to install on this machine.

Filter: All Software, Current GA, Any Edit

Available Products:

- + MySQL Servers
- + Applications
- + MySQL Connectors
- + Documentation

Products/Features To Be Installed:

- + MySQL Server 5.7.21 - X64
- + Connector/ODBC 5.3.9 - X64

Published: 2017년 7월 24일 월요일
Estimated Size: 32 MB
Release Notes: <http://dev.mysql.com/doc/relnotes/connector-odbc/en/news-5-3-9.html>

Custom > MySQL Servers > MySQL Server 5.7.21 / Connector/ODBC



MySQL 데이터 베이스

▷ MySQL 설치하기

Type and Networking

Server Configuration Type

Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.

Config Type: Development Computer

Connectivity

Use the following controls to select how you would like to connect to this server.

TCP/IP Port: 3306
 Open Windows Firewall port for network access
 Named Pipe Pipe Name: MYSQL
 Shared Memory Memory Name: MYSQL

Advanced Configuration

Select the check box below to get additional configuration pages where you can set advanced and logging options for this server instance.

Show Advanced and Logging Options



Accounts and Roles

Root Account Password

Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password: *****
Repeat Password: *****
Password strength: Weak

사용자 : root
비밀번호 : 12345

포트 : 3306

MySQL 데이터 베이스

▷ MySQL 설치 확인 및 데이터 베이스 만들기

MySQL 5.7 Command Line Client > 비번 : 12345;

```
MySQL 5.7 Command Line Client
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.21-log MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. A

Oracle is a registered trademark of Oracle Corporation and
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current
mysql> CREATE DATABASE jspdb;
Query OK, 1 row affected (0.00 sec)

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| jspdb              |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

데이터 베이스 만들기

mysql>CREATE DATABASE jspdb;

데이터 베이스 검색하기

mysql>SHOW DATABASES;



MySQL 데이터 베이스

MySQL 기본 명령어

▷ 데이터 베이스 생성하기

```
mysql> create database marketdb;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| jspdb      |
| marketdb   |
| mysql      |
| performance_schema |
| sys        |
+-----+
6 rows in set (0.00 sec)
```

데이터 베이스 만들기

CREATE DATABASE jspdb;

데이터 베이스 검색하기

SHOW DATABASES;

데이터 베이스 사용하기

USE jspdb;



MySQL 데이터 베이스

▷ 테이블 생성하기

```
mysql> CREATE TABLE user(
-> id int not null,
-> pwd varchar(20) not null,
-> name varchar(20),
-> primary key(id)
-> )default charset=utf8;
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> DESC user;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	
pwd	varchar(20)	NO		NULL	
name	varchar(20)	YES		NULL	

3 rows in set (0.00 sec)

자료형	설명
varchar(크기)	가변길이 문자열 데이터
char(크기)	고정길이 문자열
int(크기)	숫자를 저장
date	날짜 형식 저장



MySQL 데이터 베이스

▷ 테이블 조회하기

```
mysql>show tables;
```

```
mysql> use jspdb;
Database changed
mysql> show tables;
+-----+
| Tables_in_jspdb |
+-----+
| member          |
| product         |
+-----+
2 rows in set (0.00 sec)
```



MySQL 데이터 베이스

▷ 테이블 구조 변경

열(필드) 추가하기

ALTER TABLE 테이블 이름 ADD 필드이름 자료형

열(필드) 이름 변경하기

ALTER TABLE 테이블 이름 CHANGE COLUMN 기존필드 새필드 자료형

```
mysql> ALTER TABLE user ADD phone varchar(20);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> DESC user;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	
pwd	varchar(20)	NO		NULL	
name	varchar(20)	YES		NULL	
phone	varchar(20)	YES		NULL	

4 rows in set (0.00 sec)

```
mysql> ALTER TABLE user CHANGE COLUMN pwd passwd varchar(10);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> DESC user;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	
passwd	varchar(10)	YES		NULL	
name	varchar(20)	YES		NULL	
phone	varchar(20)	YES		NULL	

4 rows in set (0.02 sec)



MySQL 데이터 베이스

▷ 테이블 구조 변경

열(필드) 삭제하기

ALTER TABLE 테이블 이름 DROP 필드이름

테이블 삭제하기

DROP TABLE 테이블 이름

```
mysql> ALTER TABLE user DROP phone;
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> DESC user;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| id    | int(11) | NO  | PRI | NULL    |       |
| passwd | varchar(10) | YES |     | NULL    |       |
| name  | varchar(20) | YES |     | NULL    |       |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

```
mysql> DROP TABLE user;
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> DESC user;
ERROR 1146 (42S02): Table 'jspdbs.user' doesn't exist
```



MySQL 데이터 베이스

▷ 데이터 조작하기

데이터 삽입하기

INSERT INTO 테이블 이름(열이름1, 열이름2...) **VALUES** (데이터 값1, 데이터값 2, ...)

데이터 검색하기

SELECT 열이름1, 열이름2... **FROM** 테이블 이름

```
mysql> INSERT INTO user VALUES (100, '12345', '이양파');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM user;
+----+----+-----+
| id | pwd | name |
+----+----+-----+
| 100 | 12345 | 이양파 |
+----+----+-----+
1 row in set (0.00 sec)
```

MySQL 데이터베이스

▷ 데이터 조작하기

데이터 수정하기

UPDATE 테이블 이름 SET 열이름 1 = 데이터 값 1 [WHERE 조건식]

데이터 삭제하기

DELETE [FROM] 테이블 이름 [WHERE 조건식]

```
mysql> UPDATE user SET name='박마늘' WHERE id=100;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1    Changed: 1    Warnings: 0
```

```
mysql> SELECT * FROM user;
+----+----+-----+
| id | pwd | name |
+----+----+-----+
| 100 | 12345 | 박마늘 |
+----+----+-----+
1 row in set (0.00 sec)
```

```
mysql> DELETE FROM user WHERE id=100;
Query OK, 1 row affected (0.00 sec)
```

```
mysql> DELETE FROM user WHERE id=100;
Query OK, 0 rows affected (0.00 sec)
```

MySQL을 파이썬과 연동하기

▷ pydb 만들기

```
mysql> CREATE TABLE lang(
    -> id int PRIMARY KEY,
    -> name varchar(20) NOT NULL,
    -> description varchar(100) NOT NULL,
    -> study boolean NOT NULL DEFAULT 0
    -> )default charset=utf8;
Query OK, 0 rows affected (0.56 sec)
```

```
mysql> DESC lang;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	
name	varchar(20)	NO		NULL	
description	varchar(100)	NO		NULL	
study	tinyint(1)	NO		0	

4 rows in set (0.33 sec)

```
mysql> ALTER TABLE lang MODIFY id int NOT NULL AUTO_INCREMENT;
Query OK, 0 rows affected (0.09 sec)
```

```
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> desc lang;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
name	varchar(20)	NO		NULL	
description	varchar(100)	NO		NULL	
study	tinyint(1)	NO		0	



MySQL을 파이썬과 연동하기

▷ pymysql 모듈 사용하기

```
import pymysql

def getconn():
    conn = pymysql.connect(host='localhost', database='pydb', user='root',
                           password='12345', charset='utf8')
    return conn

print(getconn())
```



MySQL을 파이썬과 연동하기

▷ 테이블 목록 보기 - select

```
def select_lang():
    conn = getconn()
    cursor = conn.cursor()
    sql = "SELECT * FROM lang"
    cursor.execute(sql)
    rs = cursor.fetchall()
    # print(rs)
    for i in rs:
        print(i)
    conn.close()
```



MySQL을 파이썬과 연동하기

▷ 데이터 입력 - insert

```
def insert_lang():
    conn = getconn()
    cursor = conn.cursor()
    sql = "INSERT INTO lang(name, description, study) " \
          "VALUES ('PHP', 'PHP is for WEB', False)"
    cursor.execute(sql)
    conn.commit()
    conn.close()
```



MySQL을 파이썬과 연동하기

▷ 데이터 삭제 - delete

```
def delete_lang():
    conn = getconn()
    cursor = conn.cursor()
    sql = "DELETE FROM lang WHERE id=4"
    cursor.execute(sql)
    conn.commit()
    conn.close()
```



MySQL 데이터베이스

JDBC API로 데이터베이스 접속하기

```
<%@ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Database SQL</title>
</head>
<body>
<%
    Connection conn = null;
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3306/jspdb";
    String user = "root";
    String pwd = "12345";

    try{
        Class.forName(driver);
        conn = DriverManager.getConnection(url, user, pwd);
        out.println("데이터베이스 연결이 성공했습니다.");
    }catch(SQLException ex){
        out.println("데이터베이스 연결이 실패했습니다.");
        out.println("SQLException: " + ex.getMessage());
    }finally{
        if(conn !=null)
            conn.close();
    }
%>
```

localhost:8080/MysqlDB/connection.jsp

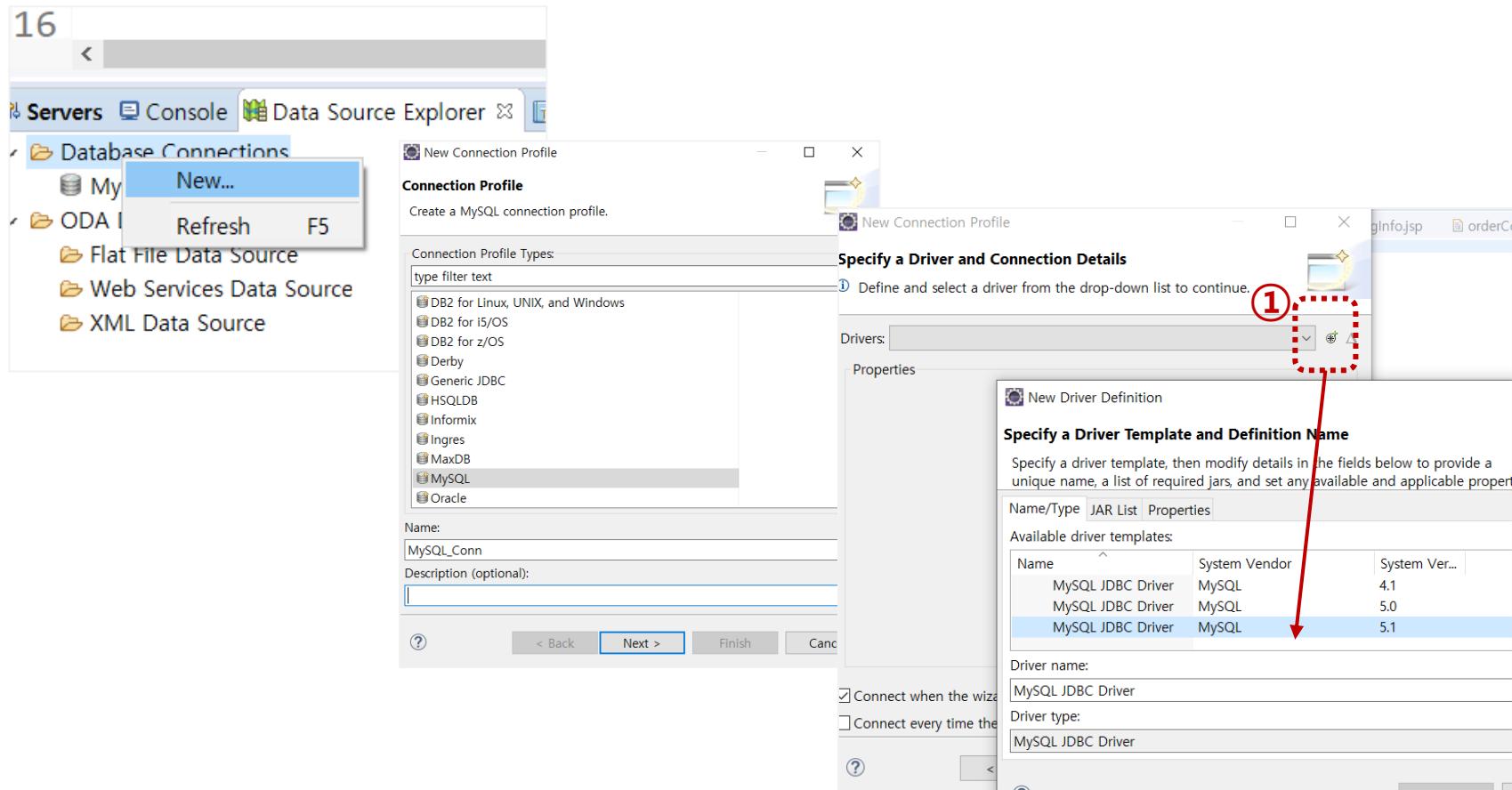
데이터베이스 연결이 성공했습니다.

데이터베이스 연결이 실패했습니다.
SQLException: Access denied for user 'root'@'localhost' (using password: YES)



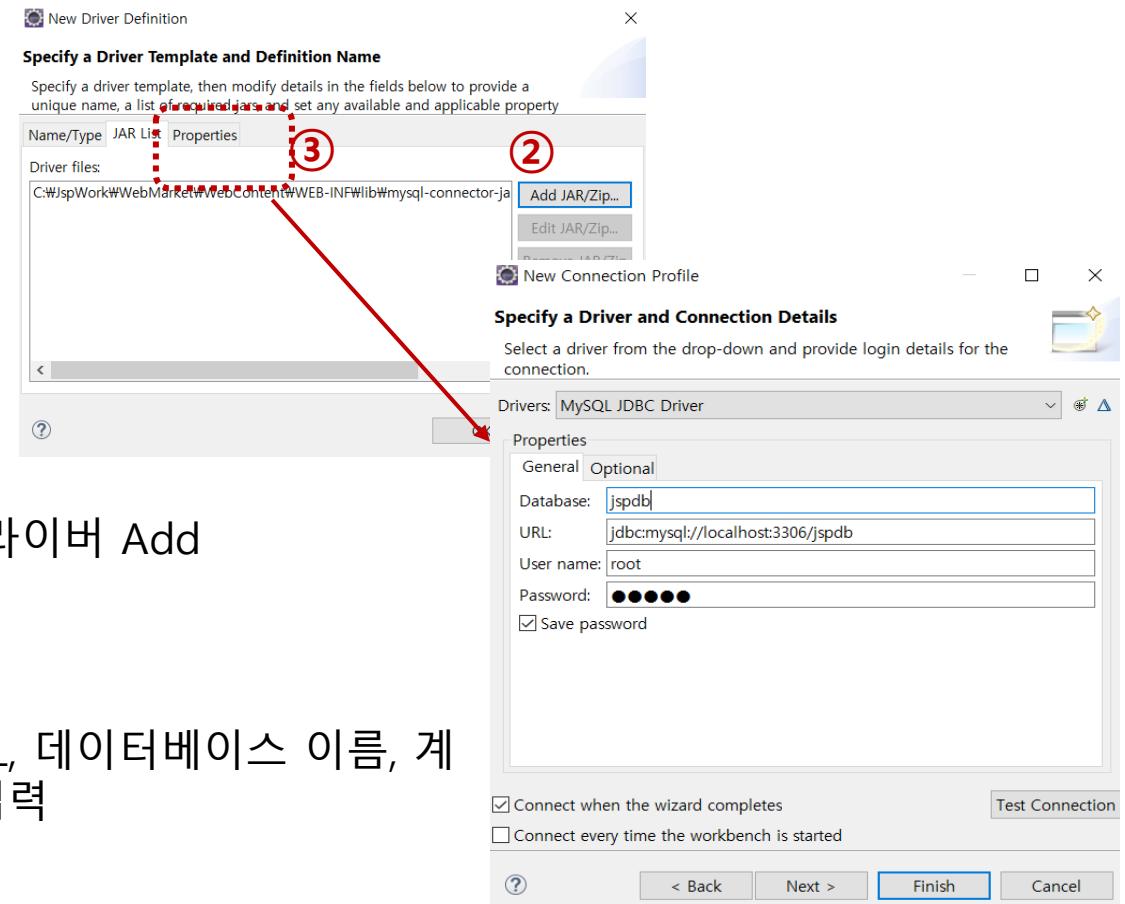
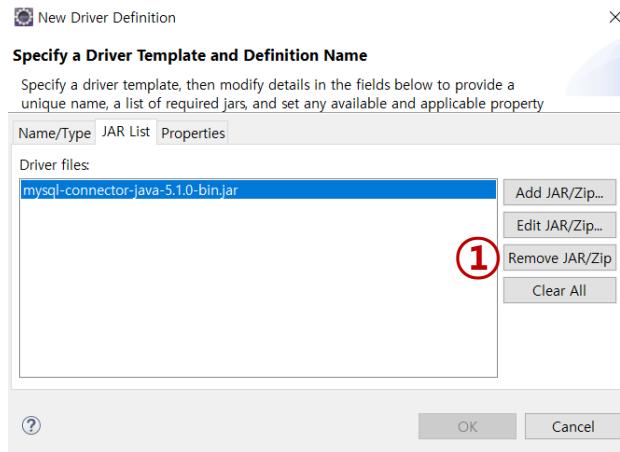
MySQL 데이터베이스

데이터베이스와 연동하기 -> 이클립스 Data Source Explorer 사용



MySQL 데이터베이스

데이터베이스와 연동하기 -> 이클립스 Data Source Explorer 사용



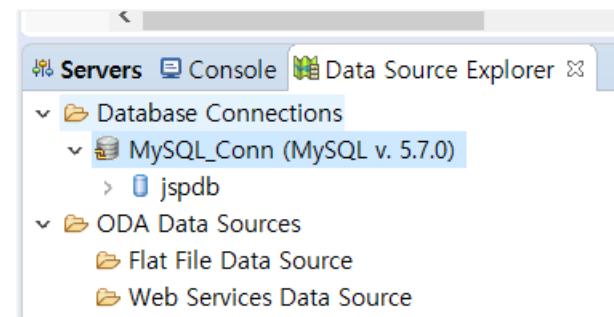
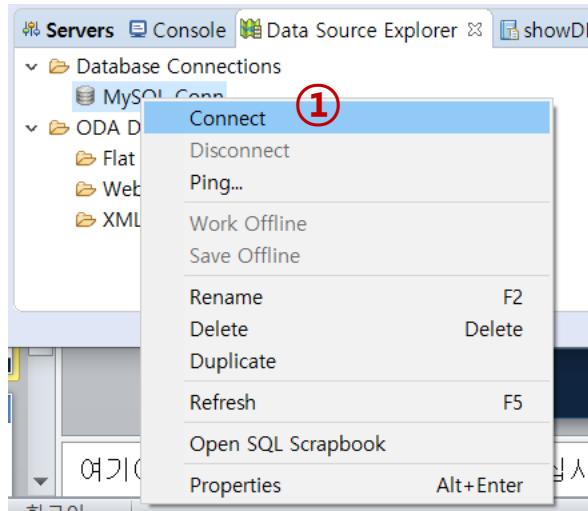
기존 드라이버 > Remove > 새 드라이버 Add

Properties 탭에서 URL, 데이터베이스 이름, 계정 이름과 비밀번호 입력

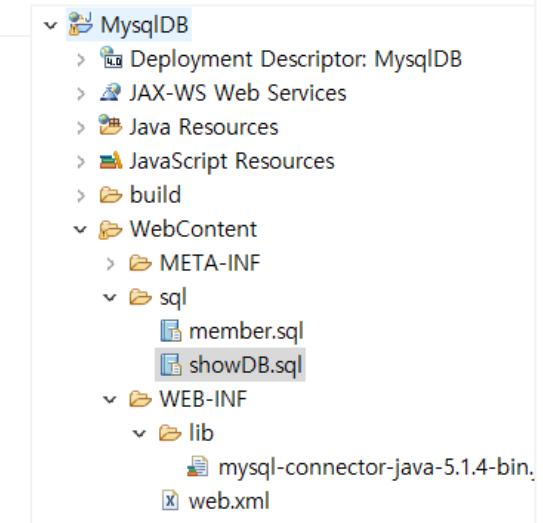


MySQL 데이터 베이스

이클립스 Data Source Explorer 사용



SQL 스크랩북 열기



DB Connect > SQL 스크랩 북 열기 >
설정 커넥션 선택(Type, Name, Database 지정)



MySQL 데이터베이스

이클립스 Data Source Explorer 사용

The screenshot shows the Eclipse IDE interface with the Data Source Explorer perspective selected. In the center, there is a SQL editor window containing the query `1 show databases;`. A context menu is open over this query, with the option `Execute Selected Text` highlighted. To the right of the editor, a results pane titled "Result1" displays the output of the query, listing the databases: information_schema, jspdb, mysql, performance_schema, and sys. A red arrow points from the "Execute Selected Text" menu item to the results pane.

Database
1 information_schema
2 jspdb
3 mysql
4 performance_schema
5 sys

명령어 선택후 Execute Selected Text > 결과확인



MySQL 데이터 베이스

jspdb에서 member 테이블 만들기



```
-- member 테이블 생성
CREATE TABLE member(
    id int NOT NULL AUTO_INCREMENT,
    name VARCHAR(20) NOT NULL,
    passwd VARCHAR(20) NOT NULL,
    PRIMARY KEY(id)
)DEFAULT CHARSET=utf8;

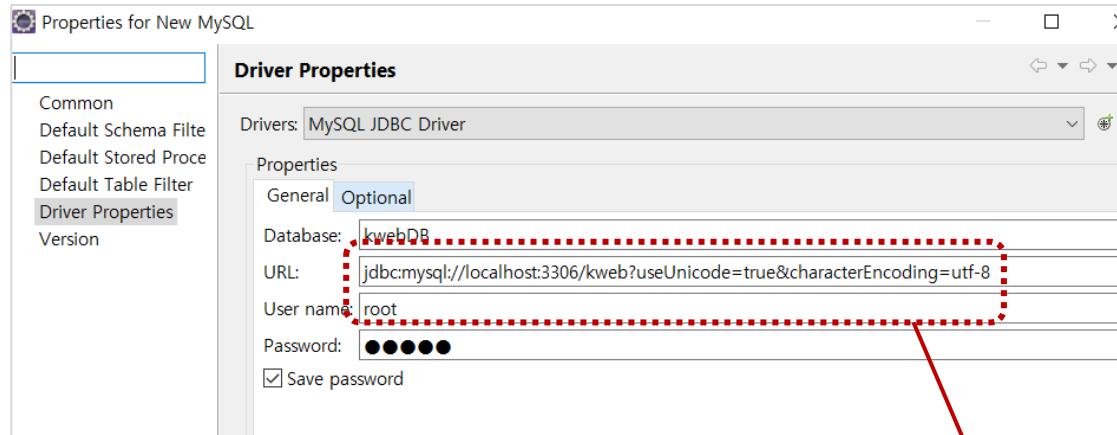
-- member 추가
INSERT INTO member VALUES (1, 'Bill', '1234');
INSERT INTO member VALUES (2, '이강', '1235');

-- member 검색
SELECT * FROM member;
```



MySQL 데이터 베이스

jspdb에서 member 테이블 만들기



한글 인코딩 설정
/kweb?useUnicode=true&characterEncoding=utf-8

	Result1		
	id	name	passwd
1	2	이강	1235
2	1	Bill	1234



MySQL 데이터베이스

jspdb에서 member 테이블 만들기

```
<%@page import="java.sql.SQLException"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.Connection"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<
    String driverClass = "com.mysql.jdbc.Driver"; //드라이버 이름
    String url = "jdbc:mysql://localhost:3306/"
        + "kweb?useUnicode=true&characterEncoding=utf-8"; //DB url
    String username = "root"; //사용자
    String password = "12345"; //비밀번호

    Connection conn = null;

    try {
        Class.forName(driverClass);
        conn = DriverManager.getConnection(url, username, password);
        out.println("데이터베이스 연결이 성공했습니다");
    } catch (SQLException e) {
        out.println("데이터베이스 연결이 실패했습니다.<br>");
        out.println("SQLException: " + e.getMessage());
    }
%>
```

dbconn.jsp

데이터베이스 연결이 성공했습니다

데이터베이스 연결이 실패했습니다.
SQLException: Access denied for user 'root'@'localhost' (using password: YES)



MySQL 데이터 베이스

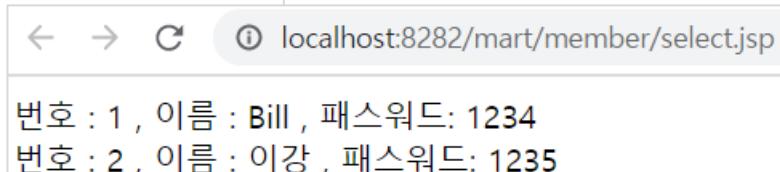
member 테이블 자료 조회하기

```
<%@ include file="dbconn.jsp" %>
<%
    PreparedStatement pstmt = null; //sql 처리 객체 선언
    ResultSet rs = null; //자료 반환 객체 선언

    try {
        String sql = "SELECT * FROM member"; //자료 조회
        pstmt = conn.prepareStatement(sql);

        rs = pstmt.executeQuery(); //쿼리 실행
        while(rs.next()) {
            out.println("번호 : " + rs.getInt("id"));
            out.println(", 이름 : " + rs.getString("name"));
            out.println(", 패스워드: " + rs.getString("passwd"));
            out.println("<br>");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally{
        if(rs != null)
            rs.close();
        if(pstmt != null)
            pstmt.close();
        if(conn != null)
            conn.close();
    }
%>
```

select01.jsp



MySQL 데이터 베이스

member 품에 데이터 삽입하기

번호 :

이름 :

패스워드 :

memberForm.jsp

```
<form action="insertProcess.jsp" method="post">
    <p>번호 : <input type="text" name="id">
    <p>이름 : <input type="text" name="name">
    <p>패스워드 : <input type="password" name="passwd">
    <p><input type="submit" value="가입">
</form>
```



MySQL 데이터 베이스

```
<%@ include file="dbconn.jsp" %>
<%
PreparedStatement pstmt = null; //sql 처리 객체 선언
//폼에 입력 자료 수집
request.setCharacterEncoding("utf-8"); //한글 인코딩
int id = Integer.parseInt(request.getParameter("id"));
String name = request.getParameter("name");
String pwd = request.getParameter("passwd");

try {
    String sql = "INSERT INTO member VALUES (?, ?, ?)"; //자료 삽입
    pstmt = conn.prepareStatement(sql);
    pstmt.setInt(1, id); //데이터 바인딩
    pstmt.setString(2, name);
    pstmt.setString(3, pwd);

    pstmt.executeUpdate(); //쿼리 실행
    out.println("자료를 추가했습니다.");
} catch (SQLException e) {
    e.printStackTrace();
} finally{
    if(pstmt != null)
        pstmt.close();
    if(conn != null)
        conn.close();
}
}
```

insertProcess.jsp

Result1			
Status	id	name	passwd
	1	Bill	1234
	2	이강	1235
	3	김기용	1236



MySQL 데이터베이스

member테이블의 데이터 수정하기

이름 : Steve -> Bill로 수정

아이디	비밀번호	이 름
a1001	1001	Steve
a1002	1002	이대리
b1001	1001	박마늘
b1002	1002	양배추



아이디	비밀번호	이 름
a1001	1001	Bill
a1002	1002	이대리
b1001	1001	박마늘
b1002	1002	양배추

아 이 디:

패스워드:

이 름:

← → ⏪ localhost:8080/MysqlDB/update01_process.jsp
Member 테이블을 수정했습니다.

MySQL 데이터베이스

member테이블의 데이터 수정하기

```
<%@ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>회원 정보 수정</title>
</head>
<body>
    <%@ include file="dbconn.jsp" %>
    <%
        request.setCharacterEncoding("utf-8");

        String id = request.getParameter("id");
        String passwd = request.getParameter("passwd");
        String name = request.getParameter("name");

        PreparedStatement pstmt = null;
        ResultSet rs = null;

        try{
            String sql = "SELECT id, passwd FROM member WHERE id=?";
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, id);
            rs = pstmt.executeQuery();
        }
    %>
```

update_process.jsp



MySQL 데이터 베이스

update_process.jsp

```
if(rs.next()){
    String rId = rs.getString("id");
    String rPasswd = rs.getString("passwd");

    if(id.equals(rId) && passwd.equals(rPasswd)){
        sql = "UPDATE member SET name=? WHERE id=?";
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, name);
        pstmt.setString(2, id);
        pstmt.executeUpdate();
        out.println("Member 테이블을 수정했습니다.");
    }else{
        out.println("비밀번호가 일치하지 않습니다.");
    }
}else{
    out.println("아이디가 일치하지 않습니다.");
}
}catch(SQLException e){
    e.printStackTrace();
}finally{
    if(rs != null)
        rs.close();
    if(pstmt != null)
        pstmt.close();
    if(conn != null)
        conn.close();
}
```

%>



MySQL 데이터베이스

member테이블의 데이터 삭제하기

아이디 : a1001 삭제

아이디	비밀번호	이 름
a1001	1001	Bill
a1002	1002	이대리
b1001	1001	박마늘
b1002	1002	양배추



아이디	비밀번호	이 름
a1002	1002	이대리
b1001	1001	박마늘
b1002	1002	양배추

아 이 디:

패스워드:



localhost:8080/MysqlDB/delete_process.jsp

Member 테이블의 자료를 삭제했습니다.

MySQL 데이터베이스

데이터베이스 쿼리 실행 – 데이터 삭제하기

```
<%@ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>회원 정보 수정</title>
</head>
<body>
    <%@ include file="dbconn.jsp" %>
    <%
        request.setCharacterEncoding("utf-8");

        String id = request.getParameter("id");
        String passwd = request.getParameter("passwd");
        String name = request.getParameter("name");

        PreparedStatement pstmt = null;
        ResultSet rs = null;

        try{
            String sql = "SELECT id, passwd FROM member WHERE id=?";
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, id);
        }
    %>
```

delete_process.jsp



MySQL 데이터베이스

```
rs = pstmt.executeQuery();
if(rs.next()){
    String rId = rs.getString("id");
    String rPasswd = rs.getString("passwd");

    if(id.equals(rId) && passwd.equals(rPasswd)){
        sql = "DELETE FROM member WHERE id=? and passwd=?";
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, id);
        pstmt.setString(2, passwd);
        pstmt.executeUpdate();
        out.println("Member 테이블의 자료를 삭제했습니다.");
    }else{
        out.println("비밀번호가 일치하지 않습니다.");
    }
}else{
    out.println("아이디가 일치하지 않습니다.");
}
}catch(SQLException e){
    e.printStackTrace();
}finally{
    if(rs != null)
        rs.close();
    if(pstmt != null)
        pstmt.close();
    if(conn != null)
        conn.close();
}
```

delete_process.jsp

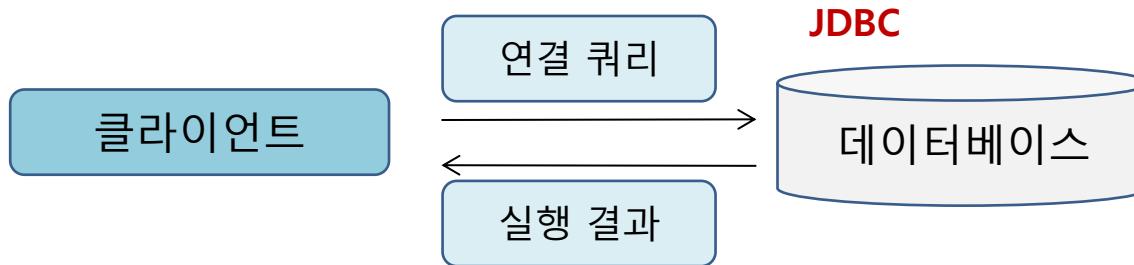


데이터베이스 커넥션 풀

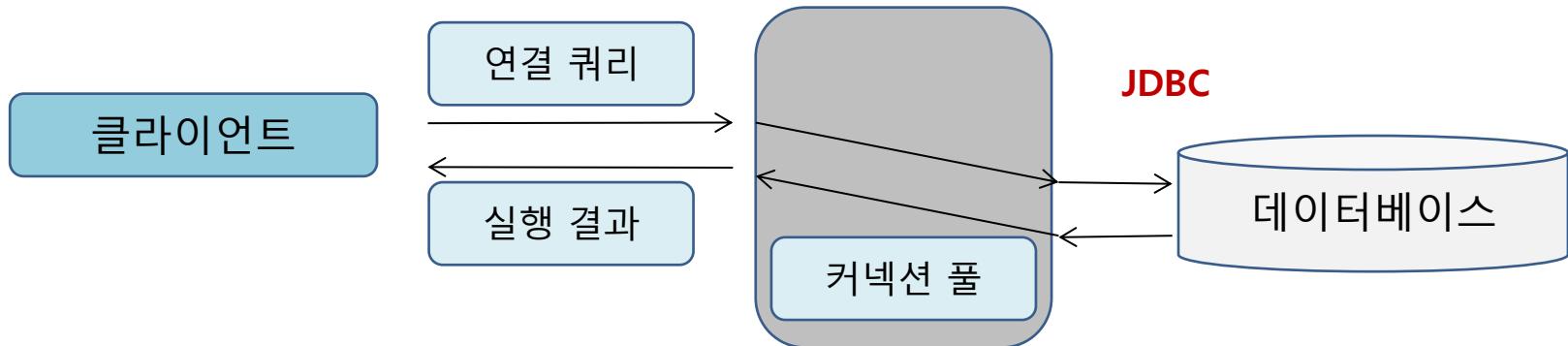
Database Connection Pool

애플리케이션에서 필요로 하는 시점에 커넥션을 만드는 것이 아니라 미리 일정한 수의 커넥션을 만들어 놓고 필요한 시점에 애플리케이션에 제공하는 서비스 및 관리 체계를 말한다.

▶ 커넥션 풀 사용하지 않음



▶ 커넥션 풀 사용



데이터베이스 커넥션 풀

커넥션 풀 동작 과정

- ① 웹 애플리케이션 서버가 시작될 때 일정 수의 커넥션을 미리 생성한다
- ② 웹 애플리케이션 요청에 따라 생성된 커넥션 객체를 전달한다.(JNDI 이용)
- ③ 일정 수 이상의 커넥션이 사용되면 새로운 커넥션을 만든다.
- ④ 사용하지 않는 커넥션은 종료하고 최소한의 기본 커넥션을 유지한다.

JNDI(Java Naming and Directory Interface)

자바에서 네이밍 서비스를 이용할 수 있도록 제공하는 인터페이스이며, 데이터베이스 커넥션 풀에서 사용하게 될 DataSource 객체는 네이밍 서비스를 통해 컨테이너에 제공한다.

- 네이밍 서비스는 논리적인 이름을 디렉터리 서비스의 파일 또는 자바 객체, 서버 등과 연결해 주는 서비스를 말하는데 DNS 서비스가 도메인 이름을 IP 숫자 주소로 변환해 주는 것과 유사한 서비스라고 생각 할 수 있다.

구체적으로는 필요한 자원을 키/값(key/value) 쌍으로 저장한 후 키를 이용해 값을 얻는 방법이다.



데이터베이스 커넥션 풀

톰캣의 DataSource 설정 및 사용 방법

- ① JDBC 드라이버(jar파일)를 이클립스의 WEB-INF/lib 폴더에 저장한다.
- ② 톰캣 Server의 context.xml 파일에 DataSource를 등록한다.
- ③ DBCP 프로젝트 만들어 연결 테스트하기

The screenshot shows the Eclipse IDE interface with two main panes. On the left, the Project Explorer pane displays a DBCP project structure:

- Deployment Descriptor: DBCP
- JAX-WS Web Services
- Java Resources
 - src
 - info.dbcptest
 - DBCPTest.java
 - Libraries
 - JavaScript Resources
 - build
- WebContent
 - META-INF
 - WEB-INF
 - lib
 - ojdbc8.jar
- web.xml

On the right, the Editor pane shows the XML configuration for the DataSource in the context.xml file:

```
<Resource  
    name="jdbc/oracle"  
    auth="Container"  
    type="javax.sql.DataSource"  
    driverClassName="oracle.jdbc.OracleDriver"  
    url="jdbc:oracle:thin:@localhost:1521:orcl"  
    username="hr"  
    password="12345"  
    maxActive="50"  
    maxWait="-1" />  
</Context>
```

Below the editor, a browser-like status bar shows the URL: localhost:8080/DBCP/dbcp. A message box at the bottom right says "DB 연결 성공!!".



데이터베이스 커넥션풀

서블릿으로 DBCP 테스트 파일 만들기

```
package info.dbcptest;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.sql.DataSource;

@WebServlet("/dbcp")
public class DBCPTest extends HttpServlet {
    private static final long serialVersionUID = 1L;
```



데이터베이스 커넥션 풀

서블릿으로 DBCP 테스트 파일 만들기

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    Connection conn;
    PreparedStatement pstmt;
    DataSource dataFactory;

    try {
        Context initContext = new InitialContext();
        Context envContext = (Context) initContext.lookup("java:/comp/env");
        dataFactory = (DataSource) envContext.lookup("jdbc/oracle");

        conn = dataFactory.getConnection();
        System.out.println("DB 연결 성공!!");

        response.setContentType("text/html; charset=utf-8");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>DB 연결 성공!! </h3>");
        out.println("</html></body>");

    } catch (NamingException e) {
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```



데이터베이스 커넥션 풀

Jsp로 커넥션 풀 구현하기 실습

프로그램 소스 목록

파일 이름	역 할
jdbc_test.sql	Jdbc_text DB 파일
jdbcTest.jsp	서블릿 파일로 작성하여 커넥션 풀 테스트

```
1 -- dbcp_test 테이블 만들기 --
2 CREATE TABLE dbcp_test(
3     name      VARCHAR2(20) NOT NULL,
4     email     VARCHAR2(30) PRIMARY KEY
5 );
6
7 SELECT * FROM dbcp_test;
```

스크립트 출력 x 질의 결과 x
SQL | 인출된 모든 행: 1(0.013초)

NAME	EMAIL
기용	kiyongee2@gmail.com



데이터베이스 커넥션 풀

Jsp로 커넥션 풀 구현하기 실습

The screenshot shows a web browser window with the URL `localhost:8080/DBCP/dbcpTest.jsp`. The page title is "이벤트 등록". It contains two input fields: "등록이름 : " and "이메일 주소: ". A "등록" button is located to the right of the email field. Below the form, the text "== 등록 목록 ==" is displayed, followed by a list of registered users:

1:기용, kiyongee2@gmail.com
2:박평순, son@korea.kr
3:박인비, inbi@test.com



데이터베이스 커넥션풀

Jsp로 커넥션 풀 구현하기 실습

```
<%>
//데이터베이스 연결 관련 변수 선언
Connection conn = null;
PreparedStatement pstmt = null;
DataSource ds = null;

try{
    Context initContext = new InitialContext();
    Context envContext = (Context)initContext.lookup("java:/comp/env");
    ds = (DataSource)envContext.lookup("jdbc/oracle");

    //커넥션 얻어오기
    conn = ds.getConnection();

    //name에 값을 입력한 경우 SQL 문장 수행 - 등록 하기
    if(request.getParameter("name") != null){
        String sql = "INSERT INTO dbcp_test VALUES (?, ?)";
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, request.getParameter("name"));
        pstmt.setString(2, request.getParameter("email"));
        pstmt.executeUpdate();
    }
} catch(Exception e){
    e.printStackTrace();
}

%>
```

dbcpTest.jsp



데이터베이스 커넥션풀

Jsp로 커넥션 풀 구현하기 실습

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>DBCP TEST</title>
<style type="text/css">
    #container{
        width: 100%;
        margin: 0 auto;
        text-align: center;
    }
</style>
</head>
<body>
    <div id="container">
        <h2>이벤트 등록</h2>
        <hr>
        <form action="dbcpTest.jsp" method="post">
            <p>등록이름 : <input type="text" name="name">
                이메일 주소: <input type="text" name="email">
                <input type="submit" value="등록">
        </form>
    </div>
```

dbcpTest.jsp



데이터베이스 커넥션풀

dbcpTest.jsp

```
<hr>
<p>== 등록 목록 ==</p>
<%
try{
    //자료 조회
    String sql = "SELECT name, email FROM dbcp_test";
    pstmt = conn.prepareStatement(sql);

    //select를 수행하면 데이터가 ResultSet의 인스턴스로 반환됨.
    ResultSet rs = pstmt.executeQuery();
    int i = 1;
    while(rs.next()){
        out.println(i + ":" + rs.getString(1) + ", " + rs.getString("email") + "<br>");
        i++;
    }
    rs.close();
    pstmt.close();
    conn.close();
}catch(Exception e){
    e.printStackTrace();
}
%>
</body>
```



트랜잭션 처리

트랜잭션이란?

데이터베이스에서 일련의 작업을 하나로 묶어 처리하는 것을 의미한다.

한 고객이 계좌이체 중 통장 A에서 통장 B로 송금을 처리하는 중에 통장 A에서는 인출되었는데 통장 B로 송금중 문제가 발생(정전, 전산오류 등)할 경우 B로 송금이 되지 않았다면 인출된 돈은 어떻게 처리해야 할까?

commit 과 rollback

Commit : 데이터 베이스에서 트랜잭션이 완료되었음을 알리는 명령이다. Commit이 완료되기 까지 결과는 현재 커넥션에서만 유효하고, 다른 커넥션에서는 처리 내용을 확인할 수 없다. 즉 commit이 완료되어야만 다른 커넥션에서도 변경된 데이터를 확인할 수 있다.

Rollback : 트랜잭션을 취소하는 명령으로, 현재 연결에서 수행한 결과를 원래대로 되돌리는 역할을 한다. 트랜잭션이 잘못되었을 경우 rollback을 이용해 모든 작업을 원래 상태로 되돌릴 수 있다.



트랜잭션 처리

삭제 했는데 이벤트 등록화면에 반영이 안되는 경우 - commit 실행

```
SELECT * FROM dbcp_test;  
  
DELETE FROM dbcp_test WHERE name='양배추';  
  
COMMIT;
```

리스트 출력 x 질의 결과 x	
SQL 인출된 모든 행: 2(0,003초)	
NAME	EMAIL
오감자	photo@test.com
박마늘	garlic@test.com

```
9  DELETE FROM dbcp_test WHERE name='양배추';  
10  
11 COMMIT;  
12
```

스크립트 출력 x | 질의 결과 x
 작업이 완료되었습니다.(0.047초)
가잇 완료.

1 행 이 (가) 삭제되었습니다.

커밋 완료.

1 행 이 (가) 삭제되었습니다.

커밋 완료.

이벤트 등록

등록이름 : 이메일 주소: 등록

-- 등록 목록 --

- 1:오감자, photo@test.com
- 2:박마늘, garlic@test.com
- 3:양배추, yang@test.com

이벤트 등록

등록이름 : 이메일 주소: 등록

-- 등록 목록 --

- 1:오감자, photo@test.com
- 2:박마늘, garlic@test.com



트랜잭션 처리

이전 명령을 취소할 경우 – rollback 실행

단, 이미 커밋이 완료되면 안됨.

The screenshot shows a sequence of SQL statements and their execution results in Oracle SQL Developer:

```
7 | SELECT * FROM dbcp_test;
8 |
9 | DELETE FROM dbcp_test WHERE name='박마늘';
10|
```

Execution results for the first part:

- 스크립트 출력 x | 질의 결과 x
- SQL | 인출된 모든 행: 1(0.002초)
- NAME EMAIL
- 1 오감자 photo@test.com

```
9 | DELETE FROM dbcp_test WHERE name='박마늘';
10|
11 ROLLBACK;
```

Execution results for the second part (immediately after the first part):

- 스크립트 출력 x | 질의 결과 x
- 작업이 완료되었습니다.(0.051초)
- 1 행 이 (가) 삭제되었습니다.
- 커밋 완료.
- 1 행 이 (가) 삭제되었습니다.
- 0개 행 이 (가) 삭제되었습니다.
- 롤백 완료.

```
9 | DELETE FROM dbcp_test WHERE name='박마늘';
10|
11 ROLLBACK;
12|
```

Execution results for the third part (immediately after the second part):

- 스크립트 출력 x | 질의 결과 x
- SQL | 인출된 모든 행: 2(0.002초)
- NAME EMAIL
- 1 오감자 photo@test.com
- 2 박마늘 garlic@test.com

회원 관리 프로젝트에 적용

회원관리 – MemberDAO.java에 적용

```
public class MemberDAO {  
  
    private Connection conn = null;  
    DataSource ds = null;  
    private PreparedStatement pstmt = null;  
    ResultSet rs = null;  
  
    //db접속  
    private void connDB() {  
        try {  
            Context initContext = new InitialContext();  
            Context envContext = (Context) initContext.lookup("java:/comp/env");  
            ds = (DataSource) envContext.lookup("jdbc/oracle");  
  
            conn = ds.getConnection();  
            System.out.println("DB 연결 성공!!");  
        } catch (NamingException e) {  
            e.printStackTrace();  
        } catch (SQLException e) {  
            e.printStackTrace();  
        }  
    }  
}
```

```
import java.sql.Connection;  
import java.sql.Date;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
  
import javax.naming.Context;  
import javax.naming.InitialContext;  
import javax.naming.NamingException;  
import javax.sql.DataSource;
```



회원 관리 프로젝트에 적용

회원추가 및 조회

회원 등록

아이디	2001
비밀번호
비밀번호 확인
이 름	이키위
성 별	<input type="radio"/> 남 <input checked="" type="radio"/> 여
등록 취소	

회원 정보

회원 가입을 축하합니다.

회원 목록

아이디	비밀번호	이름	성별	가입일	회원보기
1001	a1001	오감자	남	2020-11-08	보기
1002	a1002	양배추	남	2020-11-09	보기
1003	a1003	박마늘	여	2020-11-09	보기
2001	b2001	이키위	여	2020-11-11	보기
2002	b2002	최고추	여	2020-11-10	보기
2003	b2003	한귤	여	2020-11-10	보기

