

**Chapter 10**

**데이터베이스와 JDBC**

## JDBC의 이해

# JDBC의 개요

## ■ 기존 웹 어플리케이션 개발에서의 문제점

- 과거 소규모 웹 개발에서는 데이터를 유지하고 관리하기 위해 주로 파일을 사용했음
  - ▶ 데이터베이스가 상용화되기 전에는 데이터베이스의 비용이 너무 높았기 때문
  - ▶ 중대형 웹 어플리케이션 개발이 아닌 소규모 개발에서는 주로 파일을 사용
- 최근 대부분의 소형 웹 어플리케이션에서도 데이터베이스를 사용함
  - ▶ 데이터베이스가 상용화되고 무료로 사용할 수 있는 오픈 소스 데이터베이스들이 등장함
  - ▶ 효율적으로 데이터를 관리할 수 있게 되었고 보안에 있어서도 많은 이점을 가지게 됨
- 가장 큰 문제는 표준화(standardization)였음
  - ▶ 어플리케이션 개발을 위해 데이터베이스에 접근하는 방법이 DBMS마다 다름
  - ▶ 개발자는 데이터베이스와 연결을 위한 라이브러리를 숙지하고 있어야 함
  - ▶ DBMS를 변경할 경우 웹 어플리케이션을 구성하는 많은 문서들의 소스 코드가 수정되어야 했음

# JDBC의 개요

## ■ ODBC의 등장

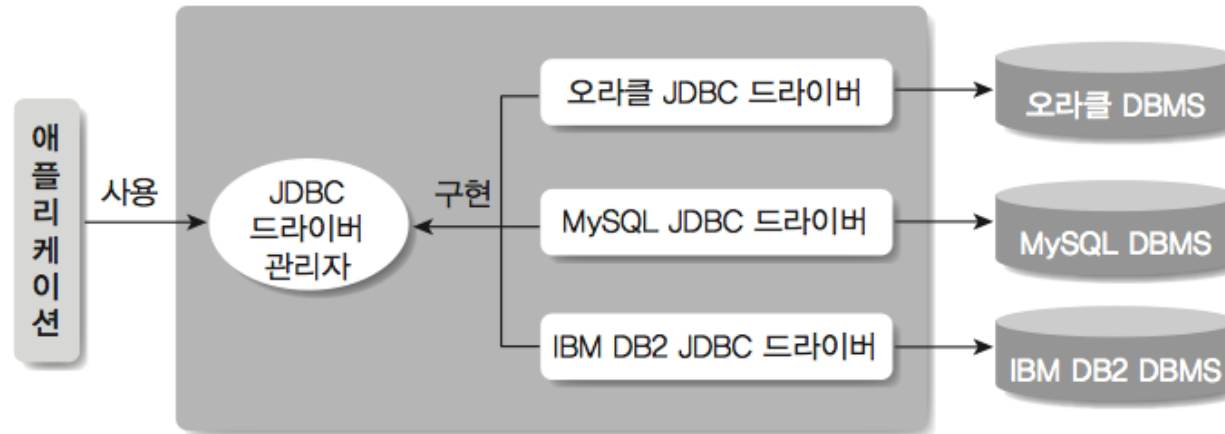
- 웹 어플리케이션과 데이터베이스의 연결을 표준화하는 방법
  - ▶ Microsoft사에서 ODBC(Open Database Connectivity)를 개발
- ODBC는 사용하고자 하는 DBMS에 관계없이 데이터베이스에 연결하기 위한 표준화된 방법을 제공
  - ▶ 개발자는 데이터베이스마다 연결을 위한 고유의 라이브러리를 알고 있어야 할 필요가 없음
  - ▶ ODBC API를 통해 표준화된 방법으로 모든 데이터베이스에 연결 가능
- 데이터베이스 벤더들이 자사의 데이터베이스를 사용할 수 있는 드라이버를 개발자에게 제공
  - ▶ 이를 ODBC 드라이버라고 함
  - ▶ 개발자는 데이터베이스에 연결하기 위한 코드를 직접 개발하지 않고 ODBC 드라이버를 사용해 표준화된 방법을 사용

# JDBC의 개요

## ■ JDBC의 등장

- ODBC는 C언어를 기반으로 개발되었음
  - ▶ Java를 기반으로 하는 개발 환경에서는 문제점을 노출하였음
- SUN Microsystem 사에서는 JDBC(Java DataBase Connectivity)를 개발
  - ▶ JDBC는 Java를 기반으로 한 개발 환경에서 데이터베이스와 연결을 위한 표준 인터페이스임
- 데이터베이스 벤더들은 JDBC API를 사용해 자사의 데이터베이스와의 연결을 위한 드라이버 제공
  - ▶ 이를 JDBC 드라이버라고 함
  - ▶ 개발자는 JDBC 드라이버를 사용해 표준화된 방법으로 데이터베이스에 연결 가능

# JDBC의 개요



- 어플리케이션은 JDBC 드라이버 관리자를 통해 연결하고자 하는 데이터베이스 드라이버를 로드
  - 데이터베이스 드라이버를 통해 데이터베이스에 접속
- 다른 데이터베이스로 포팅이 이루어진 경우 해당 데이터베이스의 드라이버만 로드하면 됨

# JDBC의 개요

## ■ 주요 DBMS에 대한 JDBC 드라이버

DBMS	JDBC 드라이버
MySQL	<code>com.mysql.cj.jdbc.Driver</code>
Oracle	<code>oracle.jdbc.driver.OracleDriver</code>
MS-SQL	<code>com.microsoft.sqlserver.jdbc.SQLServerDriver</code>

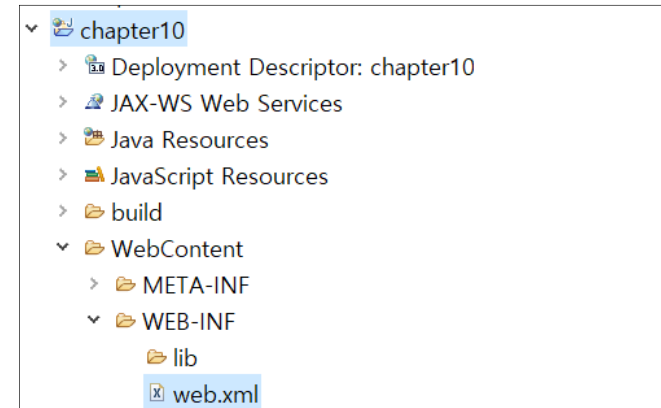
## JDBC 드라이버 다운로드와 설치



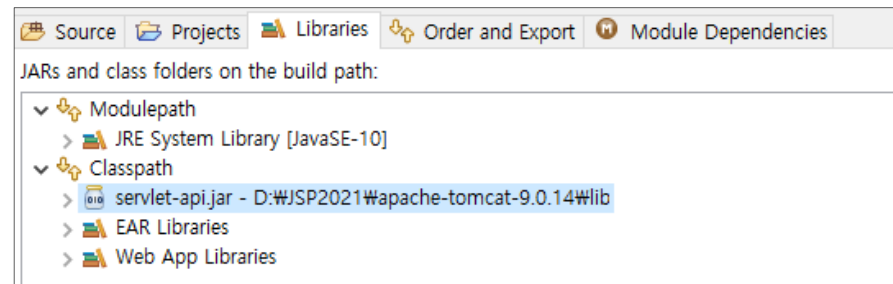
# JDBC 드라이버 다운로드

## ■ 프로젝트 생성

- 프로젝트 이름 : chapter10
  - ▶ web.xml 파일은 반드시 생성해야 함

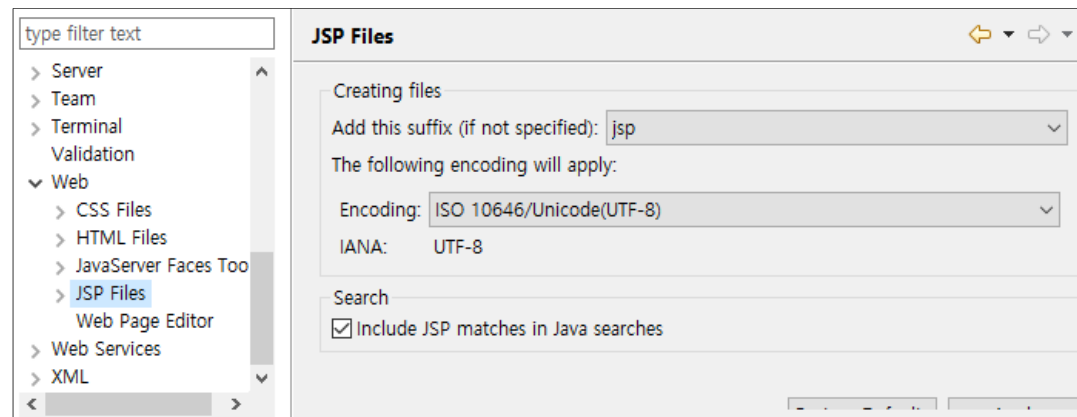


- servlet-api 라이브러리 추가



- JSP 문서 문자 집합 지정

- ▶ UTF-8로 지정



# JDBC 드라이버 다운로드

## ■ JDBC 드라이버 다운로드

- JDBC 드라이버는 각 DBMS 제조업체 홈페이지에서 무료로 다운로드 할 수 있음
  - ▶ MySQL의 경우 <https://dev.mysql.com/downloads/connector/j> 에서 다운로드 가능



# JDBC 드라이버 다운로드

- 운영체제는 Platform Independent 를 선택

▶ ZIP Archive를 다운로드함

General Availability (GA) Releases Archives

## Connector/J 8.0.23

Select Operating System:

Platform Independent

<b>Platform Independent (Architecture Independent), Compressed TAR Archive</b> (mysql-connector-java-8.0.23.tar.gz)	8.0.23	3.8M	<a href="#">Download</a>
<b>Platform Independent (Architecture Independent), ZIP Archive</b> (mysql-connector-java-8.0.23.zip)	8.0.23	4.6M	<a href="#">Download</a>

[Login »](#)  
using my Oracle Web account

[Sign Up »](#)  
for an Oracle Web account

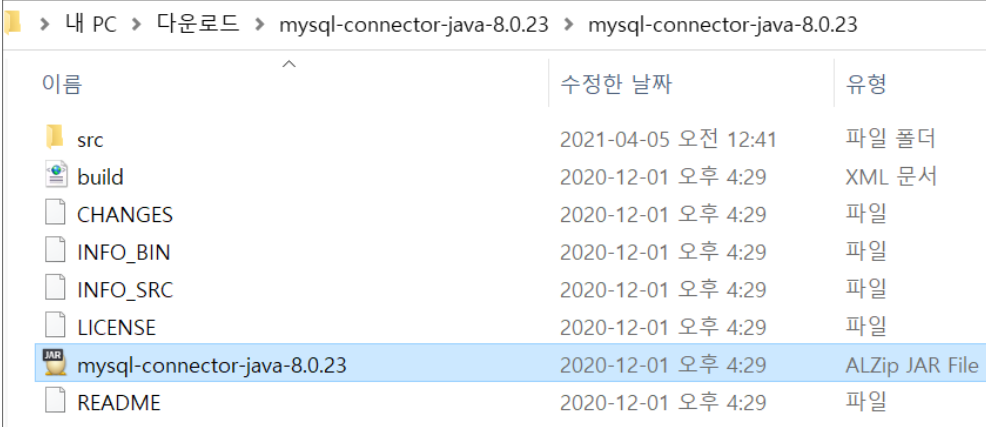
MySQL.com is using Oracle SSO for authentication. If you already have an Oracle Web account, click the Login link. Otherwise, you can signup for a free account by clicking the Sign Up link and following the instructions.

[No thanks, just start my download.](#)

# JDBC 드라이버 설치

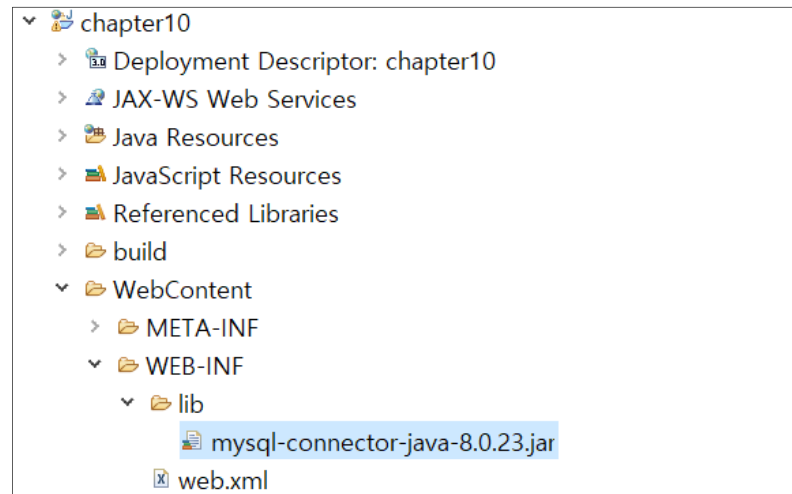
## ■ JDBC 드라이버 설치

- 압축파일을 임의의 위치에 압축 파일을 해제
  - ▶ mysql-connector-java-8.0.23.jar 파일을 복사

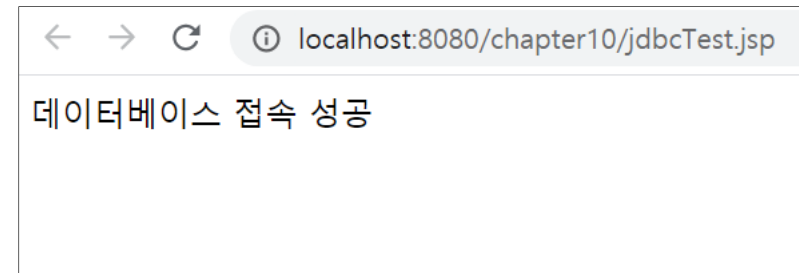


내 PC > 다운로드 > mysql-connector-java-8.0.23 > mysql-connector-java-8.0.23		
이름	수정한 날짜	유형
src	2021-04-05 오전 12:41	파일 폴더
build	2020-12-01 오후 4:29	XML 문서
CHANGES	2020-12-01 오후 4:29	파일
INFO_BIN	2020-12-01 오후 4:29	파일
INFO_SRC	2020-12-01 오후 4:29	파일
LICENSE	2020-12-01 오후 4:29	파일
mysql-connector-java-8.0.23	2020-12-01 오후 4:29	ALZip JAR File
README	2020-12-01 오후 4:29	파일

- 프로젝트의 WebContent\WEB-INF\lib에 복사



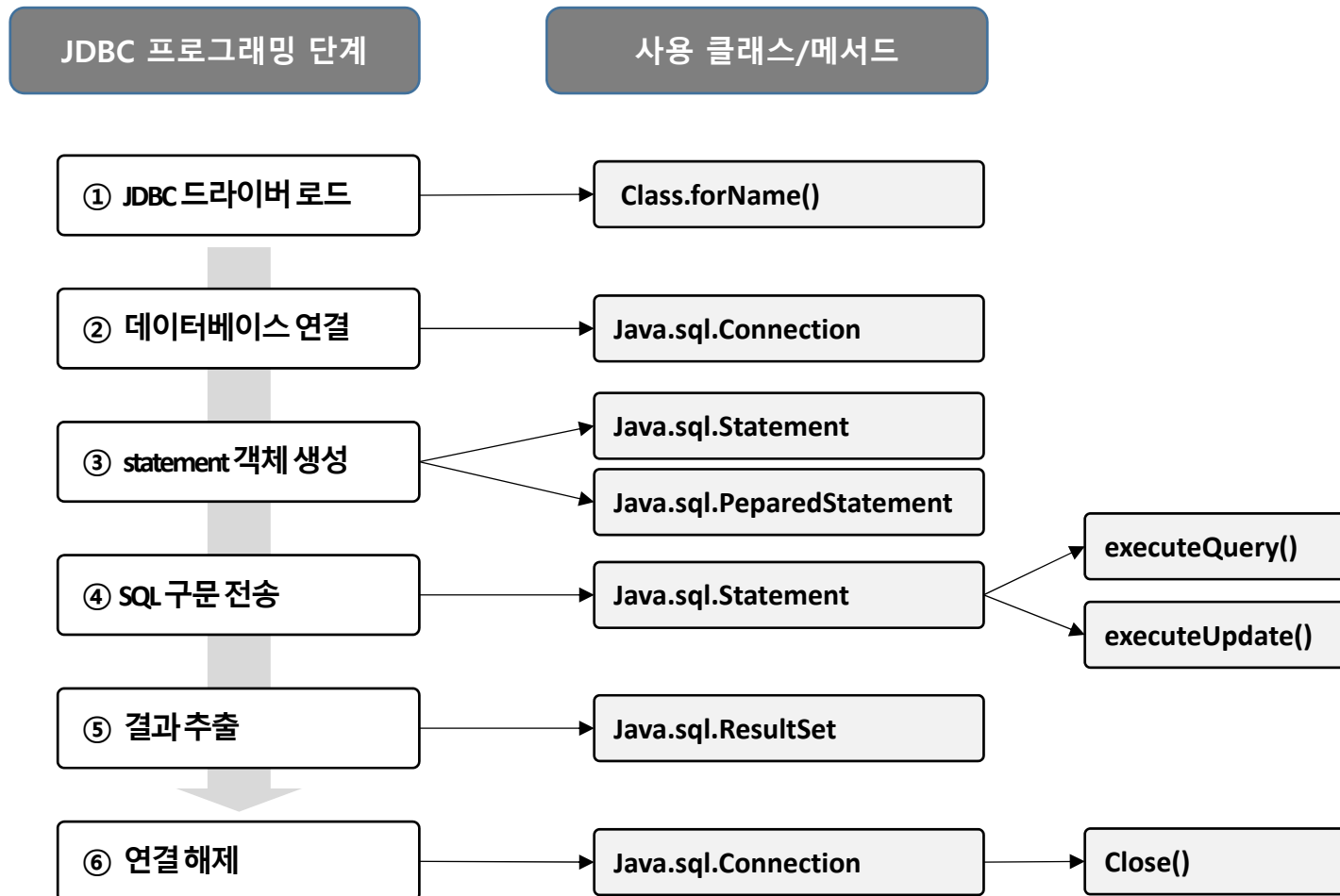
```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4
5 <%
6
7 Connection conn = null;
8
9 try {
10
11     Class.forName("com.mysql.cj.jdbc.Driver");
12     String url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC";
13     String jdbcId = "jspstudy";
14     String jdbcPw = "jsppass";
15     conn = DriverManager.getConnection(url, jdbcId, jdbcPw);
16     out.println("데이터베이스 접속 성공");
17
18 } catch (SQLException e){
19
20     out.println("데이터베이스 접속 실패");
21     e.printStackTrace();
22
23 } finally {
24     conn.close();
25 }
26
27 %>
```



## JDBC API의 이해와 활용

# JDBC 드라이버 설치

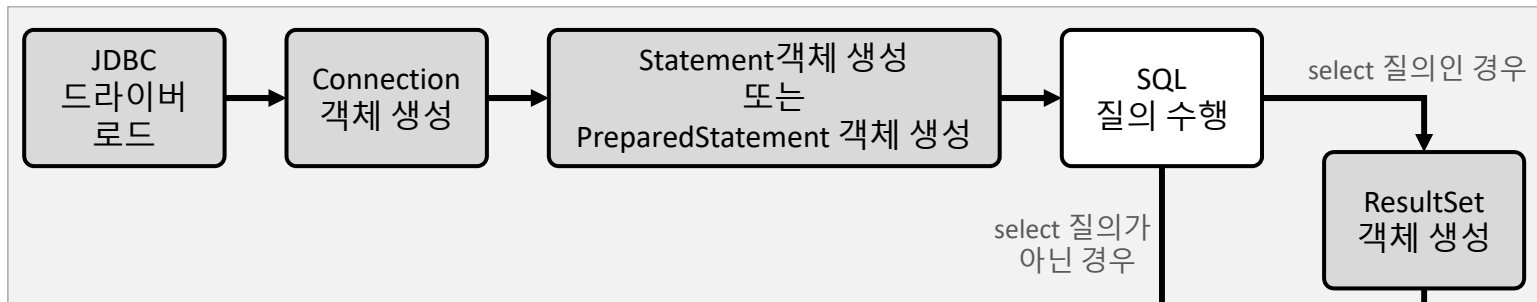
## ■ JDBC를 이용한 프로그래밍 과정



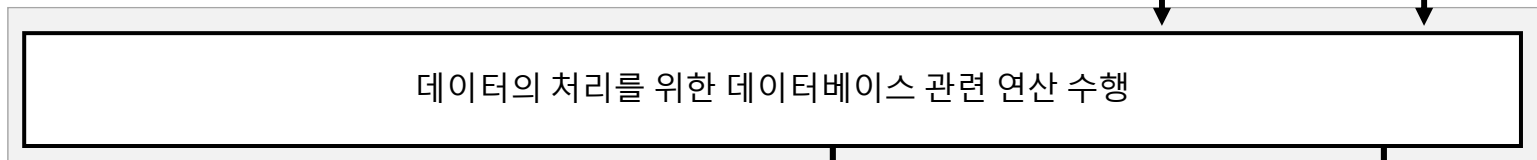
# JDBC 드라이버 설치

## JDBC를 이용한 프로그래밍 과정

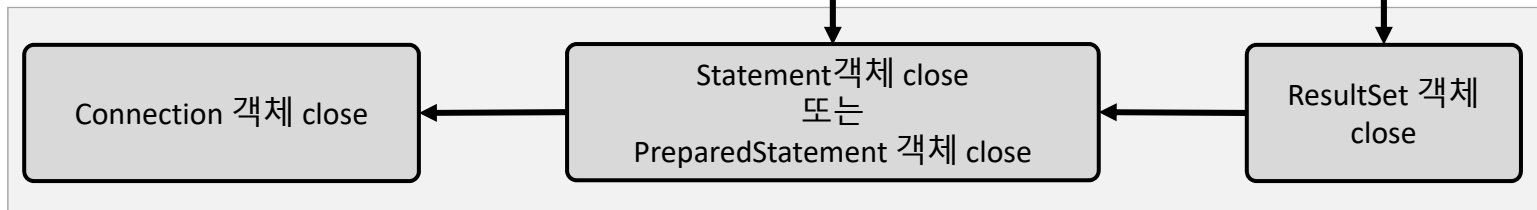
### 객체 생성 단계



### 데이터 처리 단계



### 객체 해제 단계





# 1단계 : 데이터베이스 로드

## ■ JDBC 드라이버 로드

```
Class.forName("JDBC 드라이버 이름")
```

- Class 클래스의 `forName()` 메서드 이용해 JDBC 드라이버를 로드함

- ▶ Connection 객체를 생성하기 위한 JDBC 드라이버의 로드는 JSP 문서마다 한번만 수행
  - 데이터베이스에 접근하고자 하는 모든 JSP 문서는 Connection 객체를 생성해야 함
- ▶ 드라이버가 로드되면 JDBC API를 이용해 프로그램을 작성할 상태가 된 것을 의미

- ▶ MySQL JDBC 드라이버 로드의 예

```
Class.forName("com.mysql.cj.jdbc.Driver");
```

- SQL을 사용하기 위해서는 `java.sql` 패키지를 import해야 함

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

## 2단계 : 데이터베이스 연결

### ■ Connection 객체 생성

```
Connection 객체 = DriverManager.getConnection( JDBC_URL, 아이디, 패스워드 )
```

- Connection 객체를 생성해 데이터베이스를 연결함
  - ▶ Connection 객체는 DriverManager 클래스의 getConnection() 메서드를 사용해 생성
  - ▶ getConnection() 메소드는 DriverManager 객체에 등록된 JDBC 드라이버에 연결을 수행
- getConnection() 메서드는 세 개의 인자를 가짐

인자	설명
JDBC_URL	데이터베이스 연결을 위한 정보를 포함
아이디	데이터베이스 접속을 위한 아이디
패스워드	데이터베이스 접속을 위한 패스워드

## 2단계 : 데이터베이스 연결

- **getConnection() 메서드 첫번째 인자의 형식**

- ▶ 데이터베이스에 대한 다양한 정보를 포함하고 있음

- 데이터베이스마다 다르므로 사용하고자 하는 데이터베이스 레퍼런스를 참조해야 함

- ▶ MySQL의 JDBC URL의 형식

```
jdbc:프로토콜://서버의IP주소:포트번호/스키마?serverTimezone=UTC"
```

요소	설명	지정 값
프로토콜	MySQL에서 사용하는 프로토콜을 의미	mysql
서버의 IP 주소	MySQL이 설치된 서버의 IP 주소를 의미	localhost
포트번호	MySQL이 사용하는 포트의 번호를 의미	3306
스키마	접속하고자 하는 데이터베이스의 이름을 의미	jspdb

- ▶ [예]

```
Class.forName("com.mysql.cj.jdbc.Driver");  
String url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC ";  
String jdbcId = "jspstudy";  
String jdbcPw = "jsppass";  
Connection conn = DriverManager.getConnection(url, jdbcId, jdbcPw);
```

## 3단계 : Statement 객체 생성

### ■ Statement 객체

```
Statement 객체이름 = connection객체이름.createStatement();
```

- Statement 객체는 SQL 질의를 DBMS에 전달하는 기능을 수행
  - ▶ Connection 객체의 createStatement() 메서드에 의해 생성
- MySQL에서 Statement 객체 생성의 예

```
Class.forName("com.mysql.cj.jdbc.Driver");  
String url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC";  
String jdbcId = "jspstudy";  
String jdbcPw = "jsppass";  
Connection conn = DriverManager.getConnection(url, jdbcId, jdbcPw);  
  
Statement stmt = conn.createStatement();
```

## 3단계 : Statement 객체 생성

- Statement 객체를 이용한 질의 수행

- ▶ 질의를 수행하기 위해서는 executeUpdate()나 executeQuery() 메서드를 사용
- ▶ 두 메서드의 용도와 반환 값은 서로 다름

메서드	설명
executeUpdate	SELECT 질의어가 포함되지 않는 질의의 수행에 사용 반환 타입은 int임 ( Query의 수행으로 영향을 받는 레코드의 수를 반환 ) 반드시 반환 값을 사용해야 하는 것은 아님
executeQuery()	SELECT 질의어가 포함된 질의 수행에 사용 반환 타입은 ResultSet 객체임 질의 수행 결과를 가지는 ResultSet 객체를 생성함

## 3단계 : Statement 객체 생성

- executeUpdate() 메서드의 예

- ▶ jspdb의 userinfo 테이블에서 grade 필드가 0인 값을 모두 1 증가시키는 질의 수행

```
stmt = conn.createStatement();
```

```
String Query = "UPDATE userinfo SET grade=grade+1 WHERE grade=0";  
int rows = stmt.executeUpdate(Query);
```

```
stmt = conn.createStatement();
```

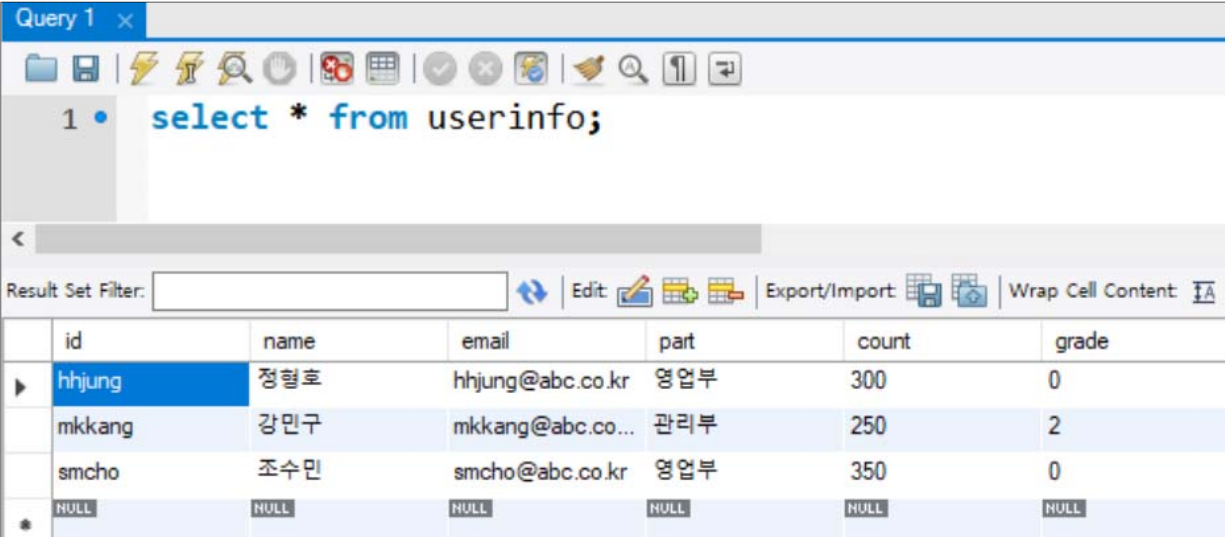
```
int rows = stmt.executeUpdate("UPDATE userinfo SET grade=grade+1 WHERE grade=0");
```

## 3단계 : Statement 객체 생성

### ■ [실습]

- jspdb의 userinfo 테이블에서 grade 필드가 0인 값을 모두 1 증가시키는 질의 수행

▶ 수행 전



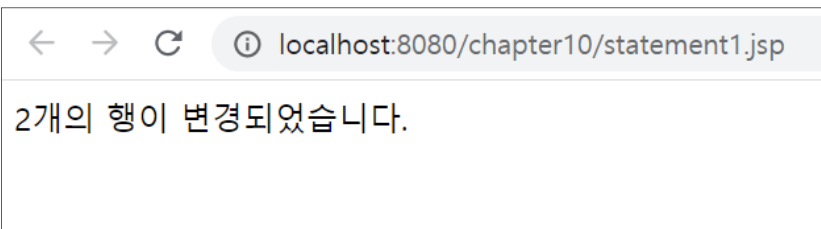
The screenshot shows a database query tool interface. At the top, there's a tab labeled 'Query 1'. Below it is a toolbar with various icons. The main area contains a SQL query: `1 • select * from userinfo;`. Below the query is a 'Result Set Filter' field. At the bottom is a table with 7 columns: id, name, email, part, count, and grade. The table contains 4 rows of data. The first row has id 'hhjung', name '정형호', email 'hhjung@abc.co.kr', part '영업부', count '300', and grade '0'. The second row has id 'mkkang', name '강민구', email 'mkkang@abc.co...', part '관리부', count '250', and grade '2'. The third row has id 'smcho', name '조수민', email 'smcho@abc.co.kr', part '영업부', count '350', and grade '0'. The fourth row has all NULL values. The first row is highlighted with a blue background.

	id	name	email	part	count	grade
▶	hhjung	정형호	hhjung@abc.co.kr	영업부	300	0
	mkkang	강민구	mkkang@abc.co...	관리부	250	2
	smcho	조수민	smcho@abc.co.kr	영업부	350	0
*	NULL	NULL	NULL	NULL	NULL	NULL

## 3단계 : Statement 객체 생성

Statement1.jsp

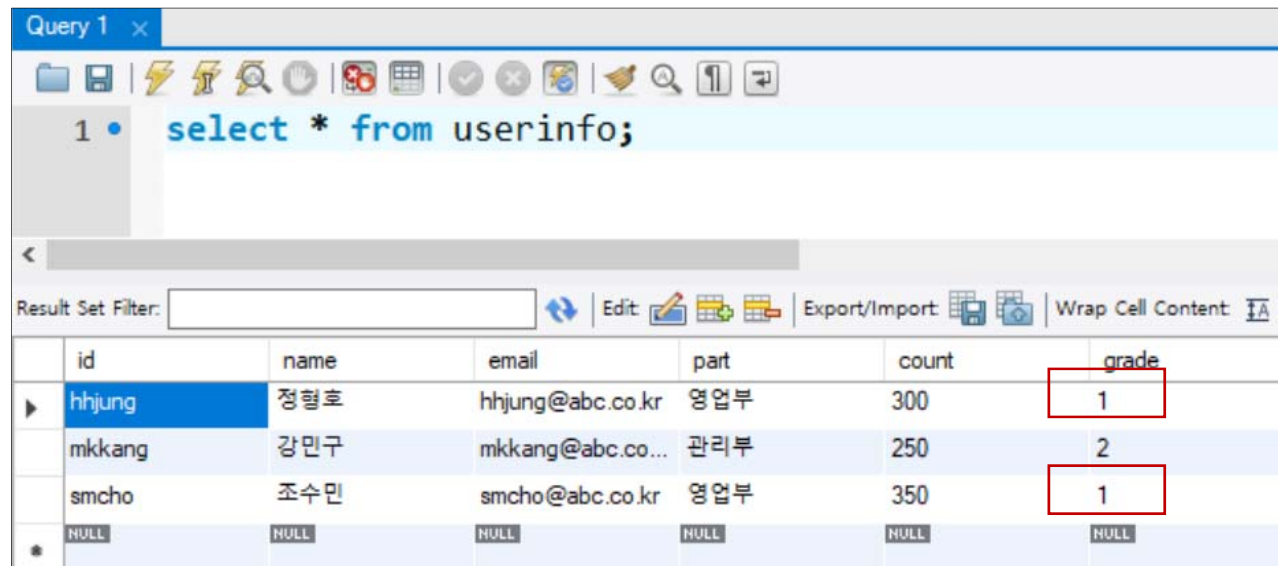
```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4 <%
5   Connection conn = null;
6
7   try {
8
9       Class.forName("com.mysql.cj.jdbc.Driver");
10      String url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC";
11      conn = DriverManager.getConnection(url, "jspstudy", "jsppass");
12
13      Statement stmt = conn.createStatement();
14      String Query = "UPDATE userinfo SET grade=grade+1 WHERE grade=0";
15      int rows = stmt.executeUpdate(Query);
16      out.println( rows + "개의 행이 변경되었습니다.");
17
18  } catch(SQLException e){
19
20      e.printStackTrace();
21
22  } finally {
23
24      conn.close();
25
26  }
27 %>
```





## 3단계 : Statement 객체 생성

▶ 실행 후



The screenshot shows a database query tool interface. At the top, a tab labeled 'Query 1' is active. Below it, a toolbar contains various icons for file operations, execution, and editing. The SQL statement 'select \* from userinfo;' is entered in the query editor. Below the editor, a 'Result Set Filter' field is visible. The main area displays a table with the following data:

	id	name	email	part	count	grade
▶	hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
	mkkang	강민구	mkkang@abc.co...	관리부	250	2
	smcho	조수민	smcho@abc.co.kr	영업부	350	1
*	NULL	NULL	NULL	NULL	NULL	NULL

## 3단계 : PreparedStatement 객체 생성

### ■ PreparedStatement 객체

```
PreparedStatement 객체이름 = connection객체이름.prepareStatement();
```

- 질의는 PreparedStatement 객체를 사용해 전달할 수도 있음
  - ▶ PreparedStatement 객체는 Connection 객체의 prepareStatement() 메서드에 의해 생성됨
- 질의를 DBMS에 전달하는 기능을 수행한다는 점에서 Statement 객체와 동일함
  - ▶ 객체를 생성하고 사용하는 방법에 있어서는 다소 차이가 있음
- SQL문을 미리 작성하고 변수를 따로 입력하는 방식
  - ▶ 질의문 내에 값을 직접 지정하지 않고 위치 홀더(position holder)를 사용
    - 위치 홀더의 값은 질의 수행 직전에 데이터형에 따른 메서드에 의해 지정됨
  - ▶ 질의문이 길 경우(입력할 데이터가 많고 종류도 다양할 경우) 효율적임
    - 입력 오류를 최소화 할 수 있음

## 3단계 : PreparedStatement 객체 생성

### ■ Statement 객체와 PreparedStatement 객체의 비교

- Statement 객체

- ▶ 질의문 내에 입력할 필드 값을 직접 표현

```
Statement stmt = conn.createStatement();  
String Query = "INSERT INTO userinfo (name, count) VALUES ('강감찬',350)";  
dbstmt.executeUpdate(Query);
```

- PreparedStatement 객체

- ▶ 입력할 값을 물음표를 사용하는 위치 홀더(position holder)로 표현

```
String Query = "INSERT INTO userinfo (name, count) VALUES (?, ?)";  
PreparedStatement pstmt = conn.prepareStatement(Query);  
pstmt.setString(1, "강감찬");  
pstmt.setInt(2, 350);  
pstmt.executeUpdate();
```

- ▶ `setString()`, `setInt()` 메서드의 첫번째 인자로 위치 홀더의 순서에 따라 일련의 번호로 지정
- ▶ 또는 일련의 번호 대신 값이 저장될 테이블의 각 필드 이름을 지정할 수도 있음

## 3단계 : PreparedStatement 객체 생성

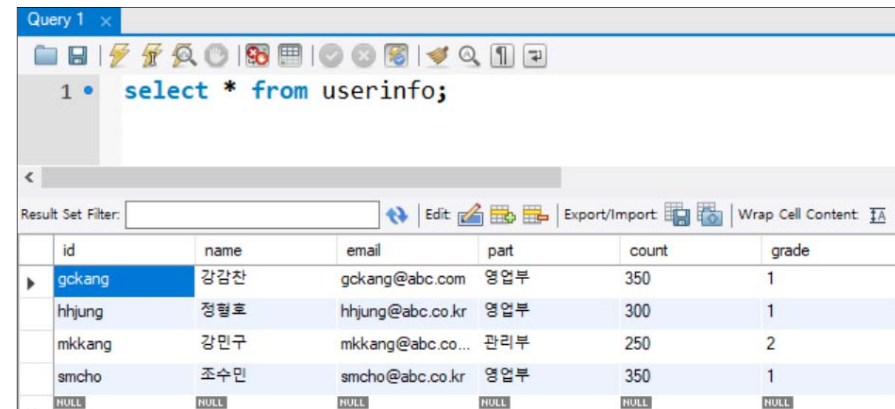
- 위치 홀더 관련 메서드

메서드	설명
setString( int parameterIndex, String x )	문자나 문자열 데이터 입력
setInt( int parameterIndex, int x )	정수형 데이터 입력
setLong( int parameterIndex, long x )	long형 데이터 입력
setDouble( int parameterIndex, double x )	double형 데이터 입력
setFloat( int parameterIndex, float x )	float형 데이터 입력
setObject( int parameterIndex, Object x )	객체 타입 데이터 입력
setDate( int parameterIndex, Date x )	Date 객체 타입 데이터 입력
setTimestamp( int parameterIndex, Timestamp x )	Timestamp형 데이터 입력

## 3단계 : PreparedStatement 객체 생성

PreparedStatement1.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4 <%
5   Connection conn = null;
6   PreparedStatement pstmt = null;
7
8   try {
9
10    Class.forName("com.mysql.cj.jdbc.Driver");
11    String url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC";
12    conn = DriverManager.getConnection(url, "jspstudy", "jspass");
13
14    String Query = "INSERT INTO userinfo VALUES (?, ?, ?, ?, ?, ?)";
15    pstmt = conn.prepareStatement(Query);
16    pstmt.setString(1, "gckang");
17    pstmt.setString(2, "강감찬");
18    pstmt.setString(3, "gckang@abc.com");
19    pstmt.setString(4, "영업부");
20    pstmt.setInt(5, 350);
21    pstmt.setInt(6, 1);
22    pstmt.executeUpdate();
23
24  } catch (SQLException e) {
25    e.printStackTrace();
26  } finally {
27    conn.close();
28  }
29 %>
```



Query 1 x

1 • select \* from userinfo;

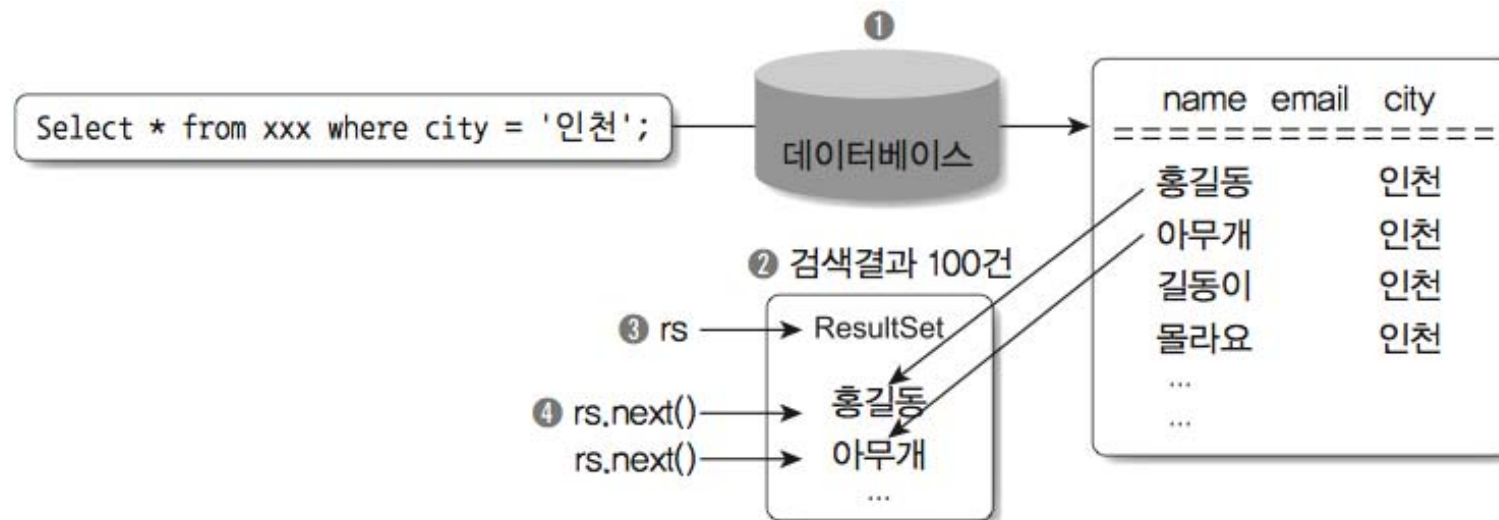
Result Set Filter: [ ] Edit Export/Import Wrap Cell Content

id	name	email	part	count	grade
gckang	강감찬	gckang@abc.com	영업부	350	1
hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
mkkang	강민구	mkkang@abc.co...	관리부	250	2
smcho	조수민	smcho@abc.co.kr	영업부	350	1
NULL	NULL	NULL	NULL	NULL	NULL

## 4단계 : ResultSet 객체 생성

### ■ ResultSet 객체

- 질의를 처리한 결과가 생성된 데이터들이 유지되는 객체
  - ▶ SQL의 질의들 중 수행된 결과가 화면에 출력되는 유일한 질의는 select 임
  - ▶ select 질의의 수행 결과는 ResultSet 객체로 반환됨
  - ▶ INSERT, UPDATE, DELETE 질의는 단순히 테이블의 내용을 갱신함으로써 수행을 종료됨
    - ResultSet 객체를 생성하지 않음
- 데이터베이스에서 조회한 결과를 받기 위해서는 executeQuery()를 사용



## 4단계 : ResultSet 객체 생성

- ResultSet 객체 생성 예

```
String Query = "SELECT part, count FROM userinfo WHERE name=?";  
PreparedStatement pstmt = conn.prepareStatement(Query);  
pstmt.setString(1, '강감찬');  
  
ResultSet rs = pstmt.executeQuery();
```

- ▶ PreparedStatement 객체를 사용해 name 필드 값이 '강감찬'인 레코드의 part 필드와 count 필드를 추출
- ▶ 검색된 결과를 포함하는 ResultSet 객체 rs를 생성

## 4단계 : ResultSet 객체 생성

### ■ ResultSet 객체의 커서(cursor)

- ResultSet 객체를 가리키는 포인터(pointer)를 의미

- ▶ ResultSet 객체를 구성하는 레코드로부터 데이터를 추출하기 위해서는 커서(cursor)를 사용해야 함
- ▶ 커서는 ResultSet 객체의 레코드를 포인트하는 포인터임
- ▶ ResultSet 객체를 구성하는 레코드가 하나 이상을 경우 커서를 이동해가며 레코드를 추출해야 함

- 커서 이동 관련 메서드

메서드	반환 타입	설명
next()	boolean	커서를 현재의 위치에서 다음 레코드로 이동시킴
previous()	boolean	커서를 현재의 위치에서 이전 레코드로 이동시킴
first()	boolean	커서를 ResultSet 객체의 맨 처음 레코드로 이동
last()	boolean	커서를 ResultSet 객체의 마지막 레코드로 이동시킴

- ▶ 이동에 성공하면 true를 반환



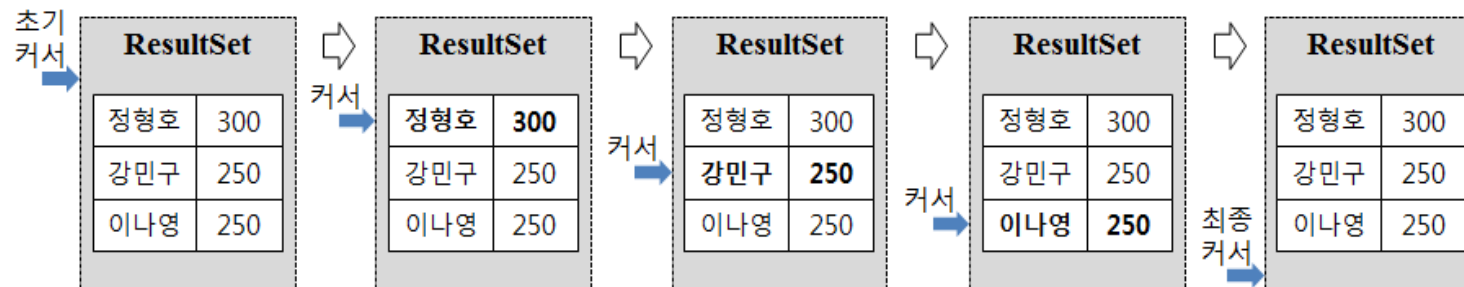
## 4단계 : ResultSet 객체 생성

### ■ 커서 사용의 예

- 커서가 ResultSet 객체의 첫 번째 레코드를 가리키는 것이 아니라 그 이전을 위치를 가리킴
  - ▶ ResultSet 객체의 첫번째 레코드를 추출하려면 next() 메서드를 수행해 커서를 첫번째 레코드로 이동해야 함

- [예]

```
select name, count from userinfo where count >=250;
```



- ▶ 커서를 다음 레코드로 이동시키기 위해 next() 메서드를 사용해야 함
- ▶ 하나의 레코드를 브라우저에 출력하고 next() 메서드를 사용해 다음 레코드로 이동
- ▶ 위의 과정을 반복하며 ResultSet 객체 내의 모든 레코드를 브라우저에 출력

## 4단계 : ResultSet 객체 생성

### ■ ResultSet 객체 내의 데이터 추출

- ResultSet 객체로 부터 추출한 레코드는 필드 단위로 출력해야 함
  - ▶ 커서가 위치한 레코드의 모든 필드를 추출한 후 다음 레코드로 커서를 옮김
  - ▶ 레코드의 각 필드에 저장된 데이터의 자료형이 다를 수 있으므로 각 자료형에 대한 데이터 추출 메서드가 필요함
- 레코드 필드 추출 메서드

메서드	설명
getString( 필드명 )	문자나 문자열 데이터 추출
getInt( 필드명 )	정수형 데이터 추출
getLong( 필드명 )	long형 데이터 추출
getDouble( 필드명 )	double형 데이터 추출
getFloat( 필드명 )	float형 데이터 추출
getObject( 필드명 )	객체형 데이터 추출
getDate( 필드명 )	Data 객체형 데이터 추출
getTimestamp( 필드명 )	Timestamp형 데이터 추출

## 4단계 : ResultSet 객체 생성

### ■ ResultSet 객체 내에 레코드가 많을 경우

- 모든 레코드를 추출하기 위해서는 for 또는 while 순환문을 사용해야 함

```
while( ResultSet객체이름.next() ) {  
  
    레코드의 데이터 추출 구문;  
    추출된 레코드의 브라우저 출력 구문;  
  
}
```

```
ResultSet rs = pstmt.executeQuery();  
  
while ( rs.next() ) {  
    String name = rs.getString("name");  
    int age = rs.getInt("email");  
    out.println("이름 : " + name);  
    out.println("이메일 : " + email);  
    out.println("<br>");  
}
```

## 5단계 : 객체의 종료와 데이터베이스 연결 해제

### ■ 객체의 종료와 데이터베이스 연결 해제

```
객체.close();
```

- 생성된 객체는 사용을 마친 다음 종료 시켜야 함
- 객체의 종료는 close() 메서드를 사용
- 객체의 종료는 생성된 순서의 역순으로 수행함
  - ▶ ResultSet 객체(생성된 경우) → Statement(또는 PreparedStatement) 객체 → Connection 객체

## 4단계 : ResultSet 객체 생성

resultSet1.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4 <%
5   Connection conn = null;
6   PreparedStatement pstmt = null;
7   ResultSet rs = null;
8
9   try {
10
11     Class.forName("com.mysql.cj.jdbc.Driver");
12     String url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC";
13     conn = DriverManager.getConnection(url, "jspstudy", "jsppass");
14
15     String Query = "SELECT * FROM userinfo";
16     pstmt = conn.prepareStatement(Query);
17     rs = pstmt.executeQuery();
18
19     out.println("<table width=560 border=1 cellspacing=0 cellpadding=2>");
20     out.println("<tr align=center>");
21     out.println("<td width=80>아이디</td>");
22     out.println("<td width=80>이름</td>");
23     out.println("<td width=200>이메일</td>");
24     out.println("<td width=80>부서</td>");
25     out.println("<td width=60>실적</td>");
26     out.println("<td width=60>등급</td>");
27     out.println("</tr>");
28
```

```
29         while(rs.next()) {
30
31             String my_id = rs.getString(1);
32             String my_name = rs.getString("name");
33             String my_email = rs.getString("email");
34             String my_part = rs.getString("part");
35             int my_count = rs.getInt("count");
36             int my_grade = rs.getInt("grade");
37
38             out.println("<tr align=center>");
39             out.println("<td>" + my_id + "</td>");
40             out.println("<td>" + my_name + "</td>");
41             out.println("<td>" + my_email + "</td>");
42             out.println("<td>" + my_part + "</td>");
43             out.println("<td>" + my_count + "</td>");
44             out.println("<td>" + my_grade + "</td>");
45             out.println("</tr>");
46
47         }
48
49         out.println("</table>");
50
51     } catch(SQLException e){
52         e.printStackTrace();
53     } finally {
54         conn.close();
55     }
56 %>
```

## 4단계 : ResultSet 객체 생성

resultSet1.jsp

아이디	이름	이메일	부서	실적	등급
gckang	강감찬	gckang@abc.com	영업부	350	1
hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
mkkang	강민구	mkkang@abc.co.kr	관리부	250	2
smcho	조수민	smcho@abc.co.kr	영업부	350	1

## 데이터베이스 활용 실습



# 실습 내용 개요

## ■ 간단한 정보 입력 및 출력

- 입력 정보

- ▶ 이름과 이메일 주소

- 입력된 정보는 데이터베이스에 저장

- 출력 내용

- ▶ 정보를 입력 받는 페이지에 출력

- 하나의 문서를 사용해 정보를 입력 받음과 동시에 출력함

## 실습 내용 개요

### 회원정보

이름  이메일

NO	NAME	EMAIL
1	이광	leeshine5455@gmail.com

### 회원정보

이름  이메일

2	홍길동	hong@abc.com
1	이광	leeshine5455@gmail.com

# 테이블 생성

## ■ 테이블 생성

### • 데이터베이스와 계정

- ▶ 아이디 : jspstudy
- ▶ 비밀번호 : jspass
- ▶ 데이터베이스 : jspdb

### • 테이블 생성

필드명	자료형	비고
userNo	int(5)	not null, primary key
userName	varchar(15)	not null
userMail	varchar(60)	not null

The screenshot shows a SQL query editor window titled 'Query 1'. The query text is as follows:

```
1 • create table member(  
2     userNo int(5) not null primary key,  
3     userName varchar(15) not null,  
4     userMail varchar(60) not null  
5 );  
6  
7 • desc member;
```

Below the query editor, there is a 'Result Set Filter' field and an 'Export' button. The result set is displayed in a table with the following columns: Field, Type, Null, Key, Default, and Extra.

Field	Type	Null	Key	Default	Extra
userNo	int(5)	NO	PRI	NULL	
userName	varchar(15)	NO		NULL	
userMail	varchar(60)	NO		NULL	

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7 </head>
8 <body>
9
10   <h2 align=center>회원정보</h2>
11   <hr align=center width=500 size=3 noshade>
12
13   <form name="member" method=post action="memberList.jsp">
14     <table align=center border=0>
15       <tr>
16         <td>이름</td>
17         <td><input type=text name="userNo" size=10 required></td>
18         <td>이메일</td>
19         <td><input type=text name="userMail" size=25></td>
20         <td><input type=submit value="SAVE"></td>
21       </tr>
22     </table>
23   </form>
24
25   <hr align=center width=500 size=2 noshade>
26   <br>
27
```

```

28< table width=400 align=center frame=hsides>
29    <tr align=left>
30        <th width=50>NO</th>
31        <th width=100>NAME</th>
32        <th width=250>EMAIL</th>
33    </tr>
34</table>
35
36< table width=400 align=center>
37    <tr>
38        <td width=50>1</td>
39        <td width=100>홍길동</td>
40        <td width=250>hong@abc.co.kr</td>
41    </tr>
42</table>
43
44</body>
45</html>
    
```

### 회원정보

---

이름 
이메일

---

NO	NAME	EMAIL
1	홍길동	hong@abc.co.kr

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4 <% request.setCharacterEncoding("utf-8"); %>
5
6 <%
7   // 데이터베이스 접속 관련 객체 선언
8   Connection conn = null;
9   Statement stmt = null;
10  PreparedStatement pstmt = null;
11  ResultSet rs1 = null;
12  ResultSet rs2 = null;
13
14  // 데이터베이스 접속 관련 변수 선언
15  String jdbc_driver = "com.mysql.cj.jdbc.Driver";
16  String jdbc_url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC";
17
18  try {
19    //JDBC 드라이버 로드와 Connection 객체 생성
20    Class.forName(jdbc_driver);
21    conn = DriverManager.getConnection(jdbc_url, "jspstudy", "jsppass");
22
23    //새로 입력되는 레코드의 userNo 값 생성 (기존의 userNo의 가장 큰 값보다 1만큼 크게 지정)
24    String Query1="select max(userNo) from member";
25    stmt = conn.createStatement();
26    rs1 = stmt.executeQuery(Query1);
27    rs1.next();
28    int new_userNo = rs1.getInt(1)+1;
```

```
29
30     //사용자의 파라미터 추출
31     String userName = request.getParameter("userName");
32     String userMail = request.getParameter("userMail");
33
34     //사용자 정보를 데이터베이스에 저장
35     String Query2 = "insert into member values (?, ?, ?)";
36     pstmt = conn.prepareStatement(Query2);
37     pstmt.setInt(1, new_userNo);
38     pstmt.setString(2, userName);
39     pstmt.setString(3, userMail);
40
41     if(userName != null) {
42         pstmt.executeUpdate();
43     }
44
45 } catch(Exception e) {
46     e.printStackTrace();
47 }
48
49 %>
50
51 <!DOCTYPE html>
52 <html>
53 <head>
54     <meta charset="UTF-8">
55 </head>
```



```

56<body>
57
58    <h2 align=center>회원정보</h2>
59    <hr align=center width=500 size=3 noshade>
60
61    <form name="member" method=post action="memberList.jsp">
62        <table align=center border=0>
63            <tr>
64                <td>이름</td>
65                <td><input type=text name="userName" size=10 required></td>
66                <td>이메일</td>
67                <td><input type=text name="userMail" size=25></td>
68                <td><input type=submit value="SAVE"></td>
69            </tr>
70        </table>
71    </form>
72
73    <hr align=center width=500 size=2 noshade>
74    <br>
75
76    <table width=400 align=center frame=hsides>
77        <tr align=left>
78            <th width=50>NO</th>
79            <th width=100>NAME</th>
80            <th width=250>EMAIL</th>
81        </tr>
82    </table>

```



```

83
84<table width=400 align=center>
85<tr>
86<td width=50>1</td>
87<td width=100>홍길동</td>
88<td width=250>hong@abc.co.kr</td>
89</tr>
90</table>
91
92</body>
93</html>

```

### 회원정보

---

이름 
이메일

---

NO	NAME	EMAIL
1	홍길동	hong@abc.co.kr

Query 1 x

1 • `select * from member;`

Result Set Filter:  Edit

	userNo	userName	userMail
▶	1	이광	leeshine5455@gmail.com
*	NULL	NULL	NULL

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4 <% request.setCharacterEncoding("utf-8"); %>
5
6 <%
7   // 데이터베이스 접속 관련 객체 선언
8   Connection conn = null;
9   Statement stmt = null;
10  PreparedStatement pstmt = null;
11  ResultSet rs1 = null;
12  ResultSet rs2 = null;
13
14  // 데이터베이스 접속 관련 변수 선언
15  String jdbc_driver = "com.mysql.cj.jdbc.Driver";
16  String jdbc_url = "jdbc:mysql://localhost:3306/jspdb?serverTimezone=UTC";
17
18  try {
19    //JDBC 드라이버 로드와 Connection 객체 생성
20    Class.forName(jdbc_driver);
21    conn = DriverManager.getConnection(jdbc_url, "jspstudy", "jsppass");
22
23    //새로 입력되는 레코드의 userNo 값 생성 (기존의 userNo의 가장 큰 값보다 1만큼 크게 지정)
24    String Query1="select max(userNo) from member";
25    stmt = conn.createStatement();
26    rs1 = stmt.executeQuery(Query1);
27    rs1.next();
28    int new_userNo = rs1.getInt(1)+1;
```

```
29
30     //사용자의 파라미터 추출
31     String userName = request.getParameter("userName");
32     String userMail = request.getParameter("userMail");
33
34     //사용자 정보를 데이터베이스에 저장
35     String Query2 = "insert into member values (?, ?, ?)";
36     pstmt = conn.prepareStatement(Query2);
37     pstmt.setInt(1, new_userNo);
38     pstmt.setString(2, userName);
39     pstmt.setString(3, userMail);
40
41     if(userName != null) {
42         pstmt.executeUpdate();
43     }
44
45 } catch(Exception e) {
46     e.printStackTrace();
47 }
48
49 %>
50
51 <!DOCTYPE html>
52 <html>
53 <head>
54     <meta charset="UTF-8">
55 </head>
```

```

56<body>
57
58    <h2 align=center>회원정보</h2>
59    <hr align=center width=500 size=3 noshade>
60
61    <form name="member" method=post action="memberList.jsp">
62        <table align=center border=0>
63            <tr>
64                <td>이름</td>
65                <td><input type=text name="userName" size=10 required></td>
66                <td>이메일</td>
67                <td><input type=text name="userMail" size=25></td>
68                <td><input type=submit value="SAVE"></td>
69            </tr>
70        </table>
71    </form>
72
73    <hr align=center width=500 size=2 noshade>
74    <br>
75
76    <table width=400 align=center frame=hsides>
77        <tr align=left>
78            <th width=50>NO</th>
79            <th width=100>NAME</th>
80            <th width=250>EMAIL</th>
81        </tr>
82    </table>

```

```
83
84= <%
85     try {
86
87         //데이터베이스로부터 사용자 정보 추출
88         String Query3 = "select * from member order by userNo DESC";
89         pstmt = conn.prepareStatement(Query3);
90         rs2 = pstmt.executeQuery();
91
92         out.println("<table width=400 align=center>");
93         //순환문을 이용해 사용자 정보 출력
94         while(rs2.next()) {
95             out.println("<tr>");
96             out.println("<td width=50>" + rs2.getInt(1) + "</td>");
97             out.println("<td width=100>" + rs2.getString(2) + "</td>");
98             out.println("<td width=250>" + rs2.getString(3) + "</td>");
99             out.println("</tr>");
100         }
101         out.println("</table>");
102
103     } catch (Exception e) {
104         e.printStackTrace();
105     }
106 %>
107 </body>
108 </html>
```

### 회원정보

---

이름  이메일

---

NO	NAME	EMAIL
1	이광	leeshine5455@gmail.com

### 회원정보

---

이름  이메일

---

NO	NAME	EMAIL
2	홍길동	hong@abc.com
1	이광	leeshine5455@gmail.com

**수고하셨습니다**