

# PreSTC 강의 – JDBC

삼성SDS

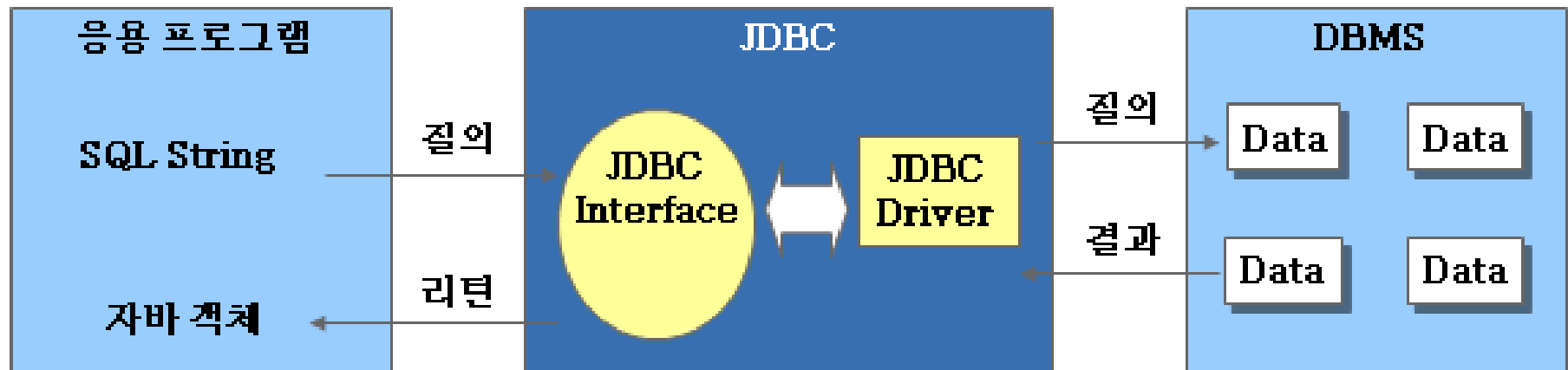


# INTRODUCTION

## □ JDBC(Java Database Connectivity)의 정의

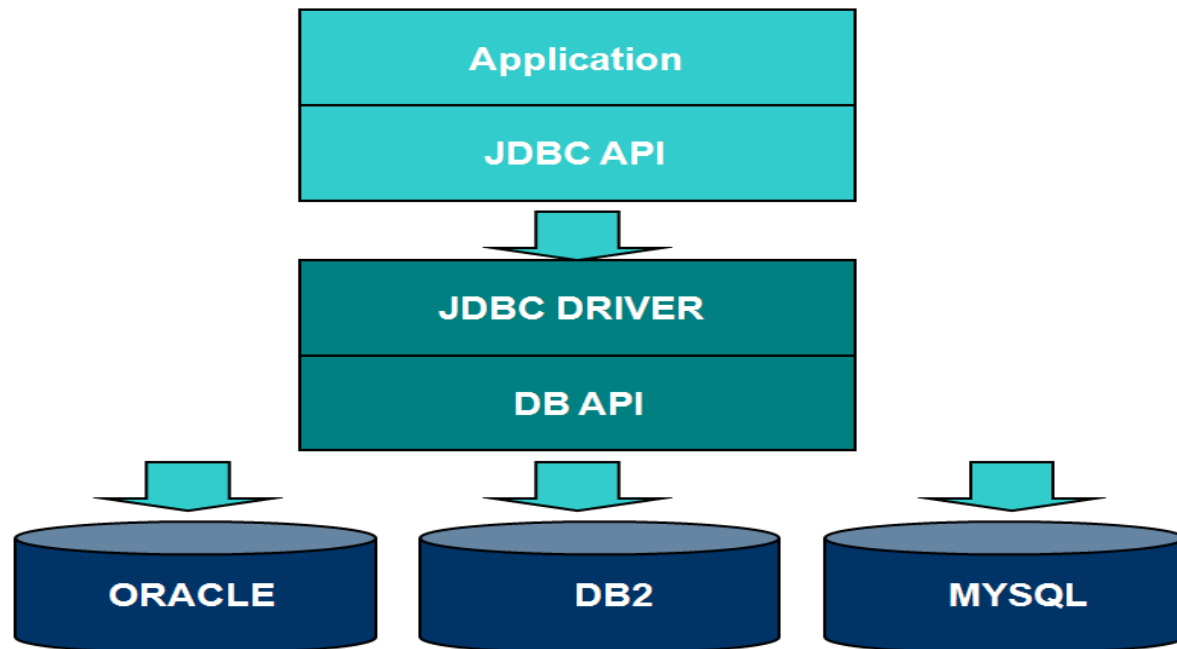
- 자바를 이용한 데이터베이스 접속과 SQL 문장의 실행, 그리고 실행 결과로 얻어진 데이터의 핸들링을 제공하는 방법과 절차에 관한 규약
- 자바 프로그램내에서 SQL문을 실행하기 위한 자바 API(java.sql 패키지)

## □ 개발자를 위한 표준 인터페이스인 JDBC API와 데이터베이스 벤더, 또는 기타 써드파티에서 제공하는 드라이버(Driver)



## □ JDBC(Java Database Connectivity)의 정의

- JDBC를 이용하면 Database에 비 종속적인 프로그램이 가능하다.
- JDBC Driver는 Database Vendor가 DB 핸들링을 위해 만들어놓은 Class 들의 집합이다.
- Oracle의 경우에는 ojdbc14.jar 파일



# JDBC 사용법

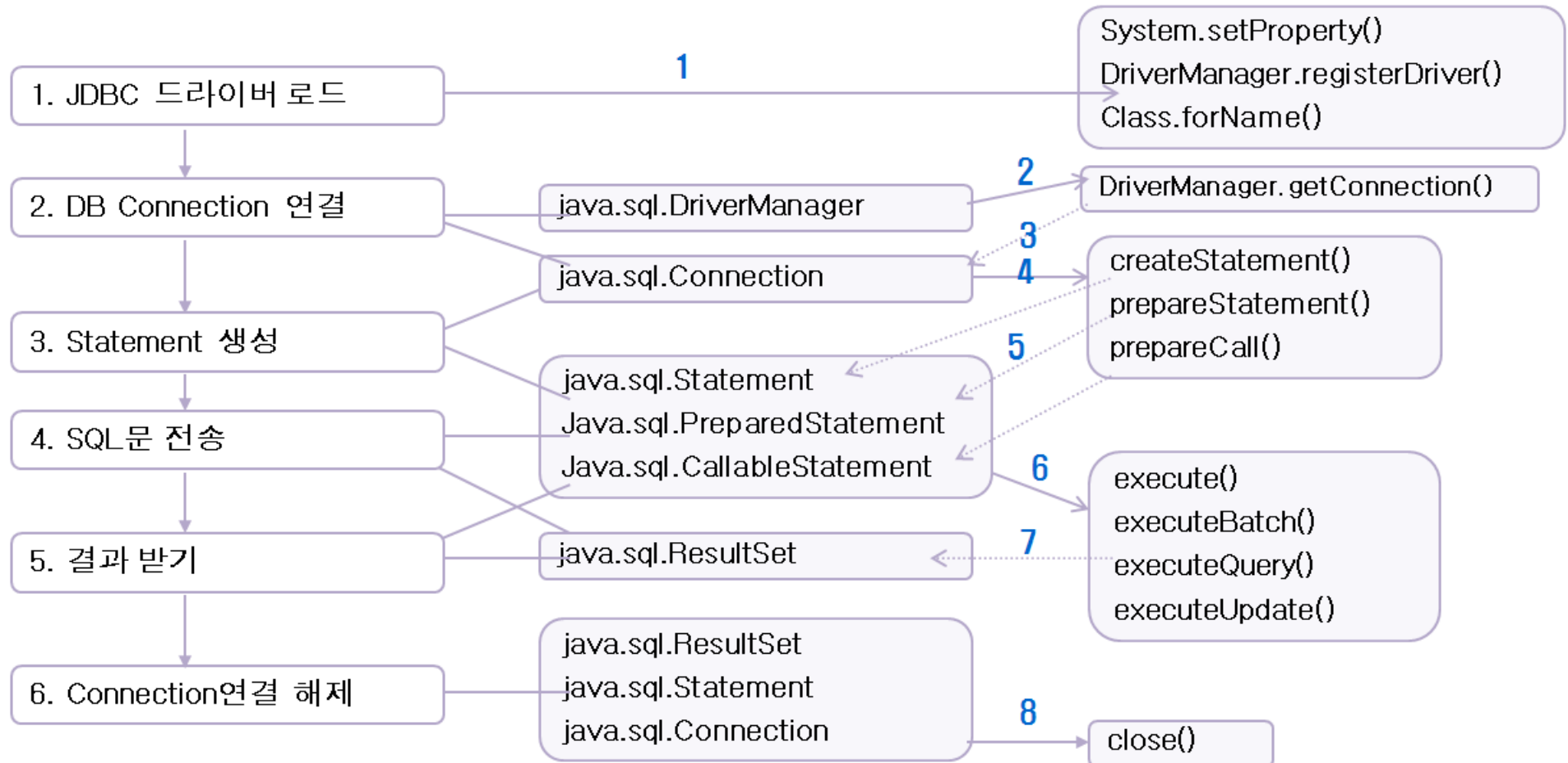
# JDBC 이용방법

## □ JDBC를 이용한 데이터베이스 연결 방법

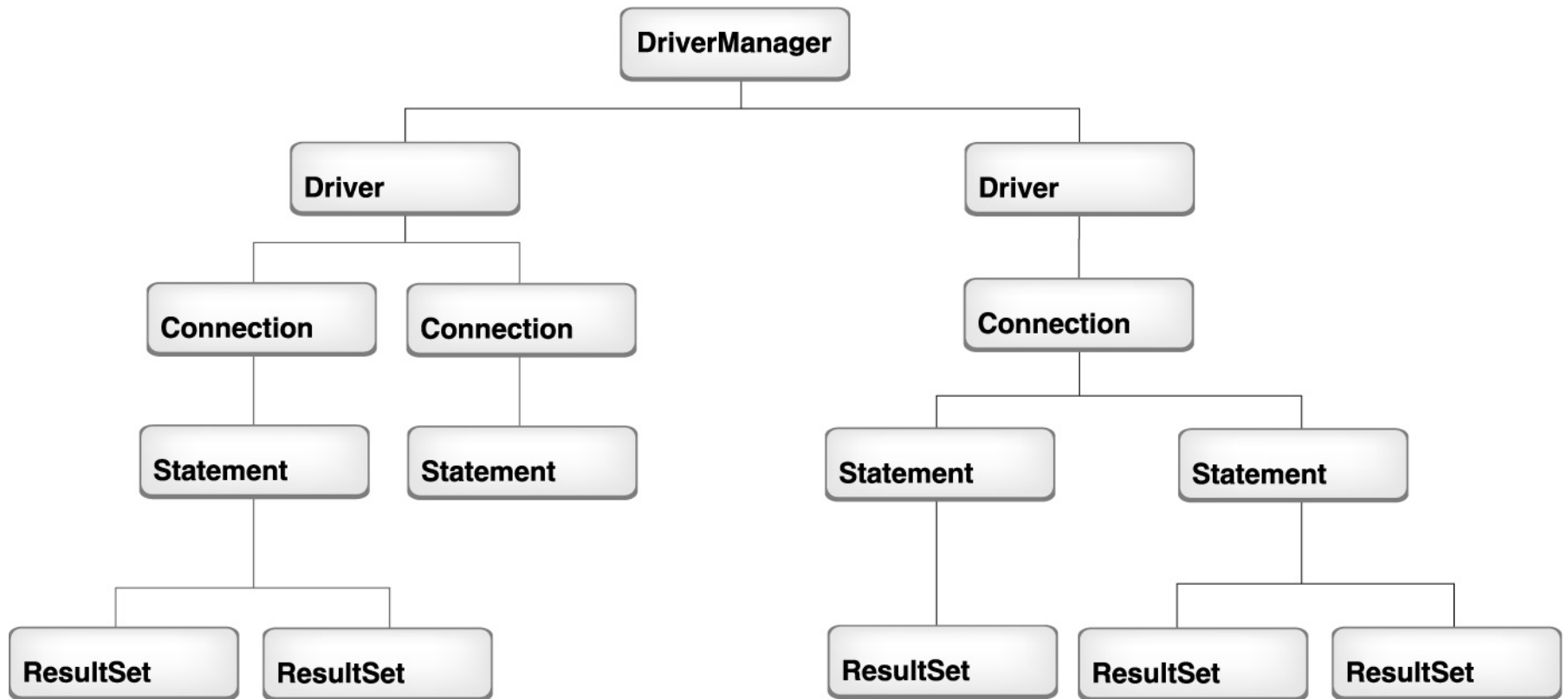
### JDBC 프로그래밍 단계

### 사용 클래스

### 수행 명령



# JDBC API 계층구조



# 단계 설명 1

## □ JDBC 드라이버 로드

```
Class.forName("oracle.jdbc.driver.OracleDriver")  
or  
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver())
```

- Oracle Driver : oracle.jdbc.driver.OracleDriver
- Informix Driver : com.informix.jdbc.IfxDriver
- MySQL Driver : org.git.mm.mysql.Driver
- SyBase Driver : com.sybase.jdbc2.jdbc.SybDriver

❖ 드라이버 로딩을 위해 각 DB 벤더에서 제공하는 Driver가 classpath에 위치해야함.



## 단계 설명 2

### □ DB Connection 연결

```
Connection conn = DriverManager.getConnection(String url)
                                   getConnection(String url Properties props)
                                   getConnection(String url, String user, String password)
```

- Oracle Driver

OCI Type : jdbc:oracle:oci7:[<userid>/<password>]@<dbname>

Thin Type : jdbc:oracle:thin:[<userid>/<password>]@<host>:<port>:<dbname>

- Informix Driver URL - jdbc:informix-sqli://<ip>:<port>/<dbname>:INFORMIXSERVER
- MySQL Driver URL - jdbc:mysql://<host>:<port>/<dbname>
- SyBase Driver - jdbc:sybase:Tds:<ip>:<port>

## 단계 설명 3-4

### □ Statement 생성 및 실행

```
Statement stmt = conn.createStatement() // Statement문 얻기
```

```
boolean isResult = stmt.execute(sql)    // 결과물이 ResultSet이면 return true  
int counts[] = stmt.executeBatch()      // Batch결과 각각 영향을 받은 수 []  
ResultSet rs = stmt.executeQuery(sql)   // 수행결과를 ResultSet으로 받음(select)  
int count = stmt.executeUpdate(sql)     // 영향받은 수 (insert, update, delete)
```

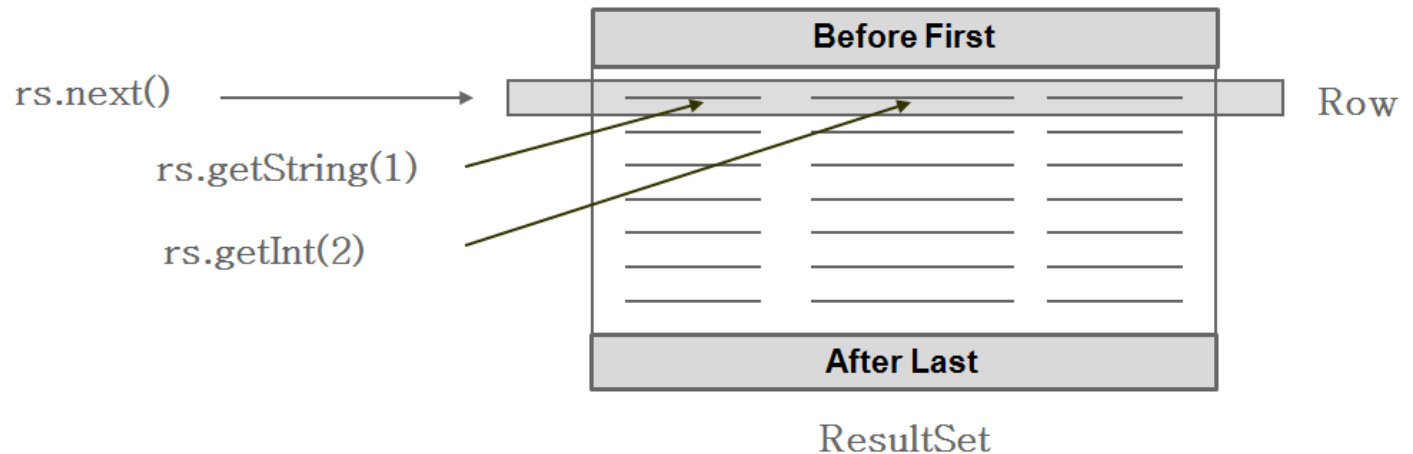
### □ PreparedStatement 생성 및 실행

```
String sql = "insert into test values(?,?)"  
PreparedStatement pstmt = conn.prepareStatement(sql)  
pstmt.setInt(1, intValue)  
pstmt.setString(2, stringValue)  
  
pstmt.execute()  
pstmt.executeBatch()  
pstmt.executeQuery()  
pstmt.executeUpdate()
```

## 단계 설명 5

### □ 결과 받기

```
ResultSet rs = null;
try{
    rs = pstmt.executeQuery();
    while(rs.next()) {
        name = rs.getString(1); // or rs.getString("name");
        age = rs.getInt(2);      // or rs.getInt("id");
    }
}catch(Exception e){
    ...
}finally{ //자원 반납하기. }
```



## 단계 설명 6

### ❑ Connection Close

```
try{
    ...
}finally{
    try{
        if(rs!=null)rs.close(); //ResultSet rs
    }catch(Exception e){
        rs = null;
    }
    try{
        if(stmt!=null)stmt.close(); // Statement stmt
    }catch(Exception e){
        stmt = null;
    }
    try{
        if(conn!=null && !conn.isClosed())conn.close(); // Connection conn
    }catch(Exception e){
        conn = null;
    }
}
```

## □ JDBCUtil 클래스 작성하기

- 워크북 Chapter3(메인 화면 및 로그인 기능 구현)에 있는 “02 로그인 기능 구현”에 있는 JDBCUtil.java 파일을 작성한다.

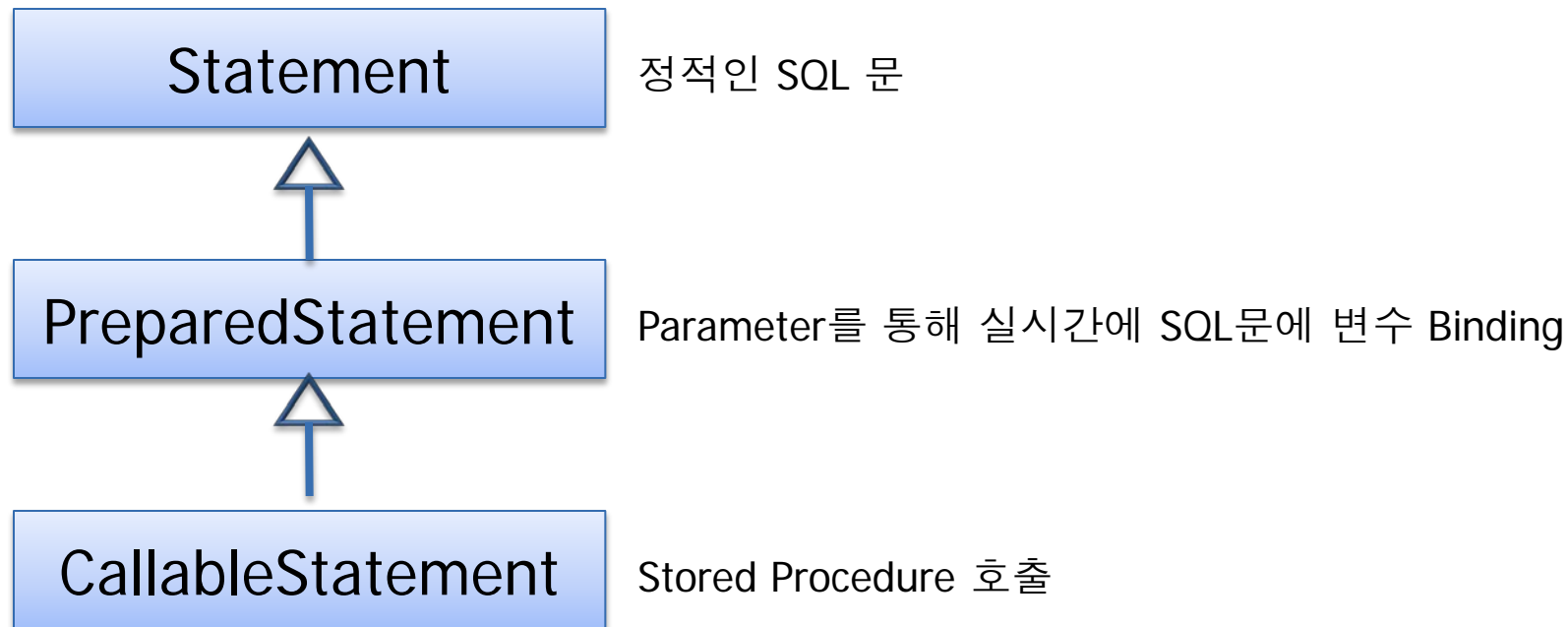
## □ JDBCUtil 클래스 테스트

- 워크북 Chapter3(메인 화면 및 로그인 기능 구현)에 있는 “02 로그인 기능 구현” ConnectionTest.java 파일을 작성하고 실행한다.

# Statement 상속관계

❑ **Statement**를 상속한 **PreparedStatement**, **CallableStatement** 클래스가 정의되어 있음

– 상속 : 부모 클래스의 기능 + alpha



# Statement

## ❑ Statement 생성

- 변수뿐 아니라 SQL문 자체가 변경될 때 사용하는 기본 Statement API
- Connection으로부터 Statement를 생성(*conn.createStatement()*)
- SQL문을 인자로 주지 않는다.

## ❑ Statement 실행

- SQL문을 실행 메소드의 인자로 준다.

stmt.executeQuery(sql) : **select 문 수행** 결과를 ResultSet에 담아서 return한다

stmt.executeUpdate(sql) : sql문의 영향을 받은 row수를 return한다.

**insert/update/delete문 수행시** 사용

- SQL문은 동일하고 변수의 값만 다른 경우라면 PreparedStatement문을 사용해야 효과적이다.

# Statement 예제

```
...  
int seq = 10;  
String sql = "SELECT * FROM BOARD WHERE SEQ = " + seq;  
PreparedStatement stmt = con.createStatement();  
ResultSet rs = stmt.executeQuery(sql);  
...
```



# PreparedStatement

## ❑ PreparedStatement 생성

- SQL문은 동일하고 변수의 값만 다른 경우에 사용한다.
- Connection으로부터 PreparedStatement를 생성(*conn.prepareStatement(sql)*)
- 수행될 SQL문을 인자로 넘겨준다, 이때 설정될 값은 ? 로 지정한다.

## ❑ PreparedStatement 실행

- ?로 표현된 변수에 값을 binding한다.
- PreparedStatement의 setXXX(변수순번, 바인딩값) method이용
- SQL문을 실행 메소드의 인자로 주지 않는다.
  - `ps.executeQuery()` : 수행결과를 ResultSet에 담아서 return한다.
  - `ps.executeUpdate()` : sql문의 영향을 받은 row수를 return한다.
- 재실행시 DBMS에서는 이미 parsing된 SQL문에 변수만 binding해서 사용한다.

## PreparedStatement 예제

```
...  
  
int seq = 10;  
  
String sql = "SELECT * FROM BOARD WHERE SEQ = ?";  
  
PreparedStatement stmt = con.prepareStatement(sql);  
  
pstmt.setInt(1, seq);  
  
ResultSet rs = stmt.executeQuery();  
  
...
```

## Lab 2.

### □ 월 별 입사자 수 출력

- 워크북 Chapter2(Project 생성 및 Database 구축)에 있는 **“03 JDBC 실습”** 에 있는

#### **3.1 월(Month) 별 입사자 수 출력하기**



Department List

localhost:8080/WebTest/employeeList1.jsp

월별 입사자 수

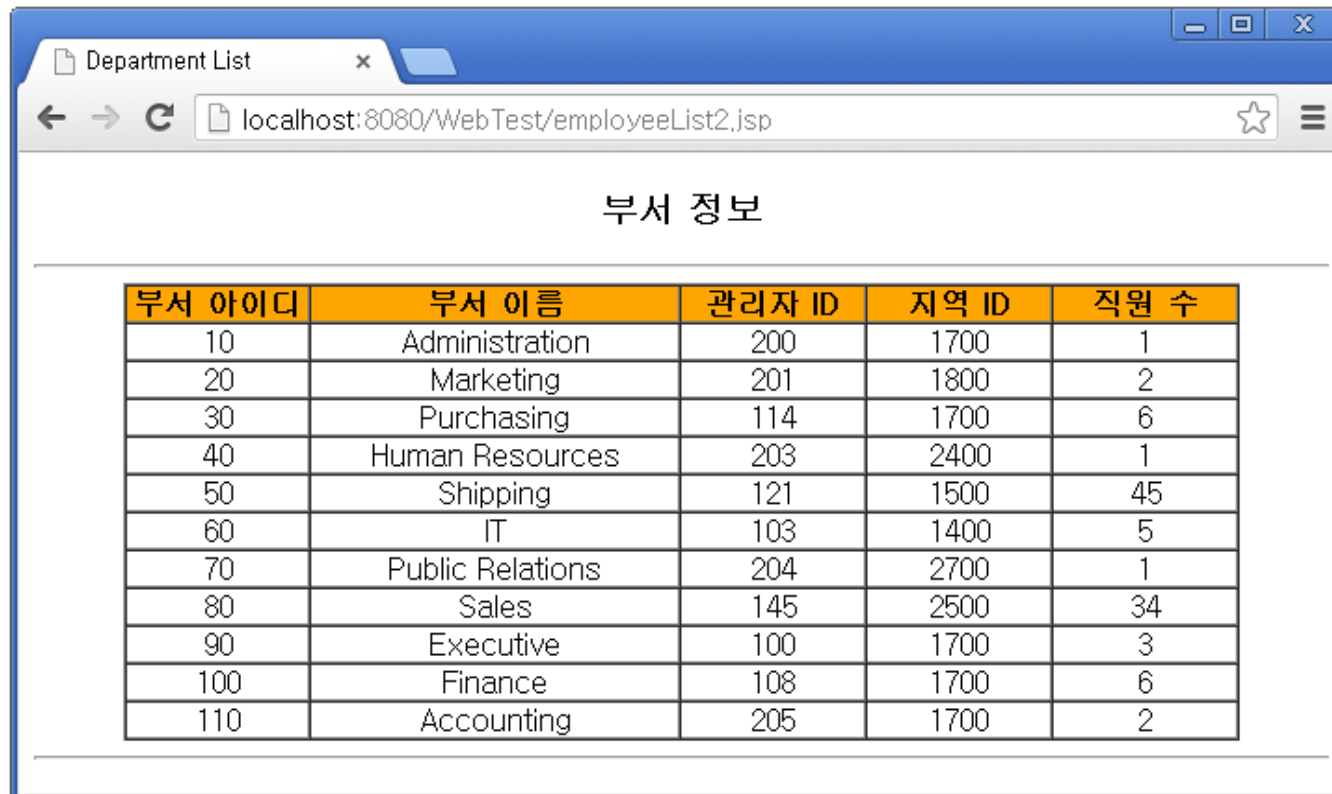
MONTH	직원수
01월	14명
02월	13명
03월	17명
04월	7명
05월	6명
06월	11명
07월	7명
08월	9명
09월	5명
10월	6명
11월	5명
12월	7명

## Lab 3.

### □ 각 부서 정보 및 직원 수 출력하기

- 워크북 Chapter2(Project 생성 및 Database 구축)에 있는 **“03 JDBC 실습”** 에 있는

#### **3.2 각 부서 정보 및 직원 수 출력하기**



The screenshot shows a web browser window with the title 'Department List'. The address bar displays 'localhost:8080/WebTest/employeeList2.jsp'. The main content area is titled '부서 정보' (Department Information) and contains a table with 5 columns: '부서 아이디' (Department ID), '부서 이름' (Department Name), '관리자 ID' (Manager ID), '지역 ID' (Region ID), and '직원 수' (Employee Count). The table lists 11 departments, including Administration, Marketing, Purchasing, Human Resources, Shipping, IT, Public Relations, Sales, Executive, Finance, and Accounting.

부서 아이디	부서 이름	관리자 ID	지역 ID	직원 수
10	Administration	200	1700	1
20	Marketing	201	1800	2
30	Purchasing	114	1700	6
40	Human Resources	203	2400	1
50	Shipping	121	1500	45
60	IT	103	1400	5
70	Public Relations	204	2700	1
80	Sales	145	2500	34
90	Executive	100	1700	3
100	Finance	108	1700	6
110	Accounting	205	1700	2

## Lab 4.

### □ 특정 부서에 근무하는 직원 검색하기

- 워크북 Chapter2(Project 생성 및 Database 구축)에 있는 **“03 JDBC 실습”** 에 있는

#### **3.3 특정 부서에 근무하는 직원 검색하기**



Employee List

localhost:8080/WebTest/employeeList3.jsp

직원 목록

부서이름

아이디	부서명	직원명	이메일	입사일	급여
114	Purchasing	Den	DRAPHEAL	1994-12-07	11000
115	Purchasing	Alexander	AKHOO	1995-05-18	3100
119	Purchasing	Karen	KCOLMENA	1999-08-10	2500
117	Purchasing	Sigal	STOBIAS	1997-07-24	2800
118	Purchasing	Guy	GHIMURO	1998-11-15	2600
116	Purchasing	Shelli	SBAIDA	1997-12-24	2900