# 데이터베이스와 JDBC

# 데이터베이스 준비

## ❖ 회원 테이블 준비

```
create table member (
 name varchar2(10),
 userid varchar2(10),
 pwd varchar2(10),
 email varchar2(20),
 phone char(13),
 admin number(1) default 0, -- 0:사용자, 1:관리자
 primary key(userid)
);
```

# 데이터베이스 준비

## ❖ 회원 테이블 준비

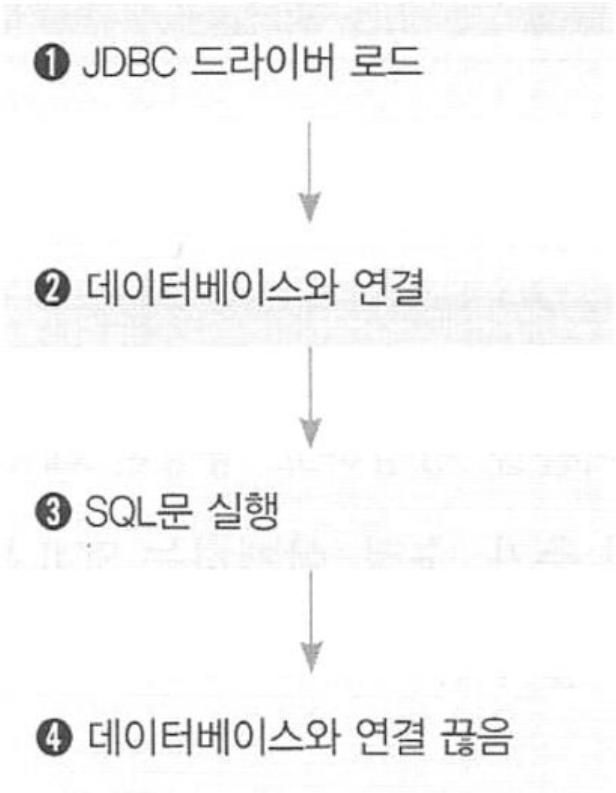
```
insert into member values('이소미', 'somi', '1234', 'gmd@naver.com',
'010-2362-5157', 0);
insert into member values('하상오', 'sang12', '1234', 'ha12@naver.
com','010-5629-8888', 1);
insert into member values('김윤승', 'light', '1234', 'youn1004@naver.
com','010-9999-8282', 0);


commit;
```

# JDBC를 이용한 데이터 조작하기

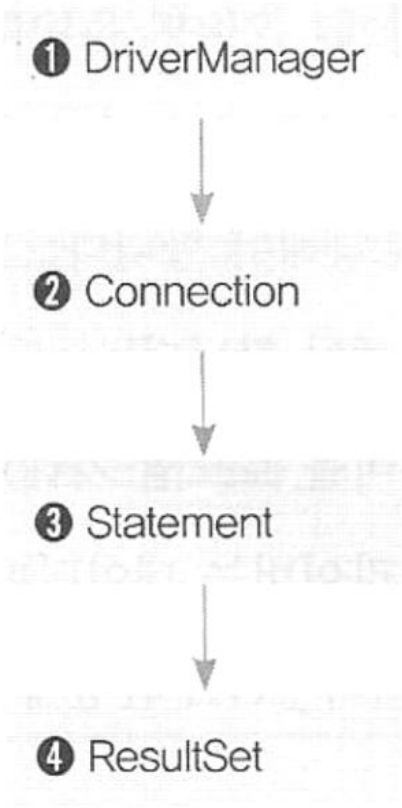
## ❖ JDBC

- Java Database Connectivity
- 데이터베이스 프로그래밍 API



# JDBC를 이용한 데이터 조작하기

## ❖ SELECT 쿼리문 실행



## ❖ JDBC 객체

```
import java.sql.*;
```

- 인터페이스
  - Connection
    - 데이터베이스 연결,
    - DriverManger.getConnection()으로 획득
  - Statement :
    - 쿼리
    - Connection의 createStatement() 메서드
  - ResultSet : SELECT 쿼리 결과 인터페이스
    - Statement의 executeQuery()의 리턴값

# JDBC를 이용한 데이터 조작하기

## ❖ 데이터베이스와 연결하기

- JDBC 드라이버 로딩

```
Class.forName(JDBC 드라이버 클래스명);
```

```
Class.forName("oracle.jdbc.driver.OracleDriver");
```

- DriverManger 로 데이터베이스에 연결(로그인)
  - 연결에 성공하는 경우 Connection 객체 리턴
  - 실패하는 경우 예외 발생

```
Connection DriverManager.getConnection(접속URL, 사용자ID, 비밀번호);
```

  - 접속 URL
    - jdbc:oracle:thin:[호스트명][:포트번호]:sid

```
"jdbc:oracle:thin:@localhost:1521:XE"
```

# JDBC를 이용한 데이터 조작하기

## ❖ 데이터베이스와 연결하기

- DriverManger 로 데이터베이스에 연결(로그인)

```
Conn conn = DriverManger.getConnection(
                "jdbc:oracle:thin:@localhost:1521:XE",
                "scott",
                "tiger");
```

## ❖ 데이터베이스 연결 끊기(로그아웃)

```
conn.close()
```

## ❖ SELECT 문과 Statement, ResultSet 클래스

- Statement
  - SQL 쿼리 실행을 위한 객체
  - Connection 인스턴스의 createStatement() 메서드로 생성

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

  - 작업 완료후 닫기

```
stmt.close();
```

## ❖ SELECT 문과 Statement, ResultSet 클래스

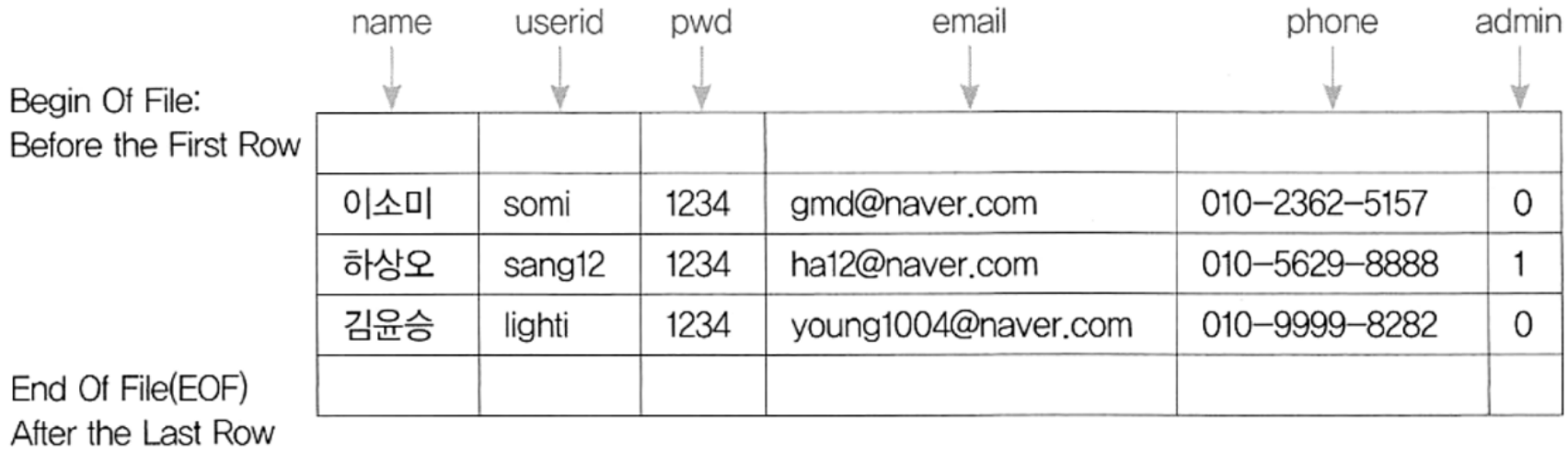
- 쿼리 실행

| 메소드 | 설명 |
|---|---|
| executeQuery | select 문과 같이 결과값이 여러 개의 레코드로 구해지는 경우에 사용한다. |
| executeUpdate | insert, update, delete 문과 같은 내부적으로 테이블의 내용이 변경만 되고 결과 값이 없는 경우에 사용한다. |

```
String sql ="select * from member";
ResultSet rs = stmt.executeQuery(sql);
```

## ❖ SELECT 문과 Statement, ResultSet 클래스

- ResultSet 클래스
  - SELECT 쿼리 실행 결과를 저장



## ❖ SELECT 문과 Statement, ResultSet 클래스

- ResultSet 클래스

| 메소드 | 설명 |
|---|---|
| next() | 현재 행에서 한행 앞으로 이동 |
| previouse() | 현재 행에서 한행 뒤로 이동 |
| first() | 현재 행에서 첫 번째 행의 위치로 이동 |
| last() | 현재 행에서 마지막 행의 위치로 이동 |

  - 성공시 true 리턴, 실패시 false 리턴

## ❖ SELECT 문과 Statement, ResultSet 클래스

- ResultSet 클래스
  - 결과 순회

```
String sql = "select * from member";
ResultSet rs = stmt.executeQuery(sql);
while(rs.next()){

}
```

| Begin Of File: Before the First Row | | | | | |
|---|---|---|---|---|---|
| 이소미 | somi | 1234 | gmd@naver.com | 010-2362-5157 | 0 |
| 하상오 | sang12 | 1234 | ha12@naver.com | 010-5629-8888 | 1 |
| 김윤승 | lighti | 1234 | young1004@naver.com | 010-9999-8282 | 0 |
| End Of File(EOF) After the Last Row | | | | | |

**Cursor의 위치**

rs.next()

## ❖ SELECT 문과 Statement, ResultSet 클래스

- ResultSet 클래스
  - 컬럼 데이터 추출
    - get데이터타입(컬럼명 또는 컬럼 인덱스)

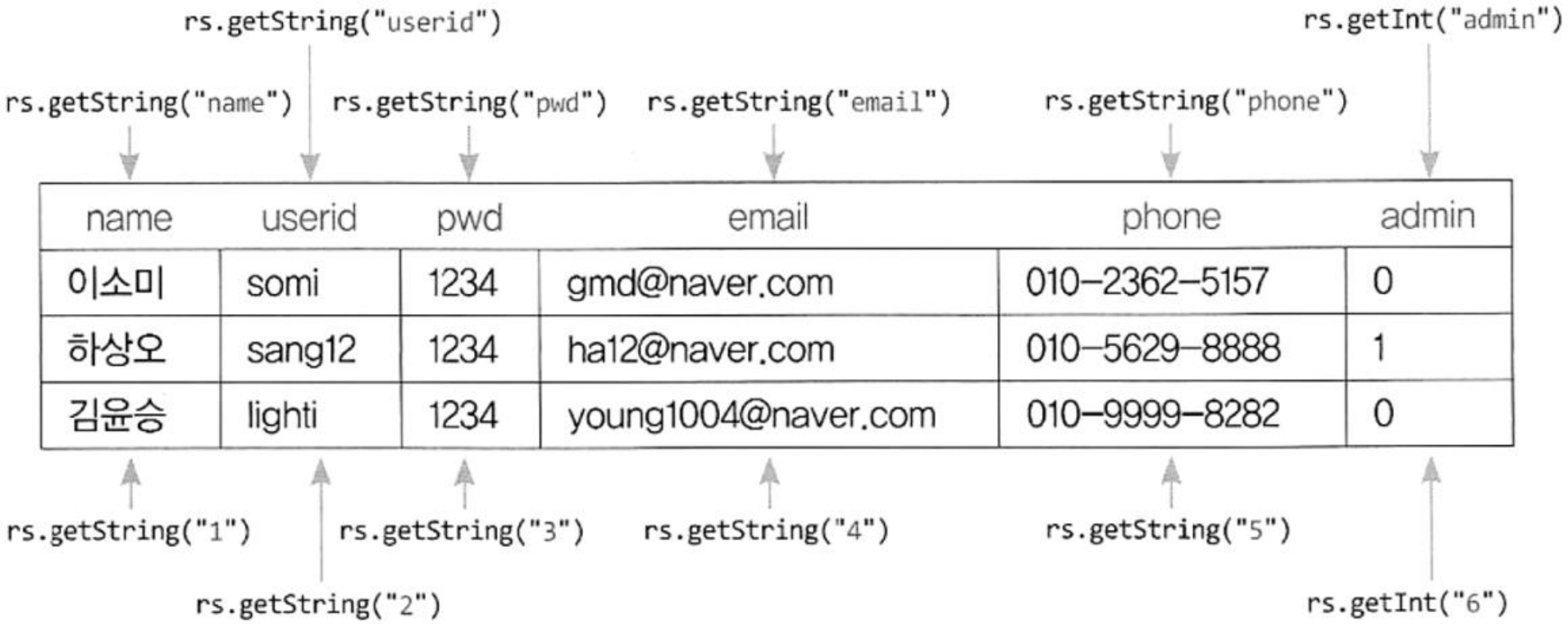
```
int admin = rs.getInt("admin");
String name = rs.getString("name");
```

```
while(rs.next()){
      out.println(rs.getString("name"));
      out.println(rs.getString("userid"));
      out.println(rs.getString("pwd"));
      out.println(rs.getString("email"));
      out.println(rs.getString("phone"));
      out.println(rs.getInt("admin"));
}
```

```
while(rs.next()){
      out.println(rs.getString(1));
      out.println(rs.getString(2));
      out.println(rs.getString(3));
      out.println(rs.getString(4));
      out.println(rs.getString(5));
      out.println(rs.getInt(6));
}
```

## ❖ SELECT 문과 Statement, ResultSet 클래스

- ○ ResultSet 클래스
  - ▪ 컬럼 데이터 추출



## ❖ 01_allMember.jsp

```
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%!//선언부는 첫 방문자에 의해서 단 한번 수행합니다.
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String url = "jdbc:oracle:thin:@localhost:1521:XE";
    String uid = "scott";
    String pass = "tiger";
    String sql = "select * from member";%>
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Insert title here</title>
</head>
```

## ❖ 01_allMember.jsp

```
<body>
    <table width='800' border='1'>
        <tr>
            <th>이름</th>
            <th>아이디</th>
            <th>암호</th>
            <th>이메일</th>
            <th>전화번호</th>
            <th>권한(1:관리자, 2:일반회원)</th>
        </tr>
```

## ❖ 01_allMember.jsp

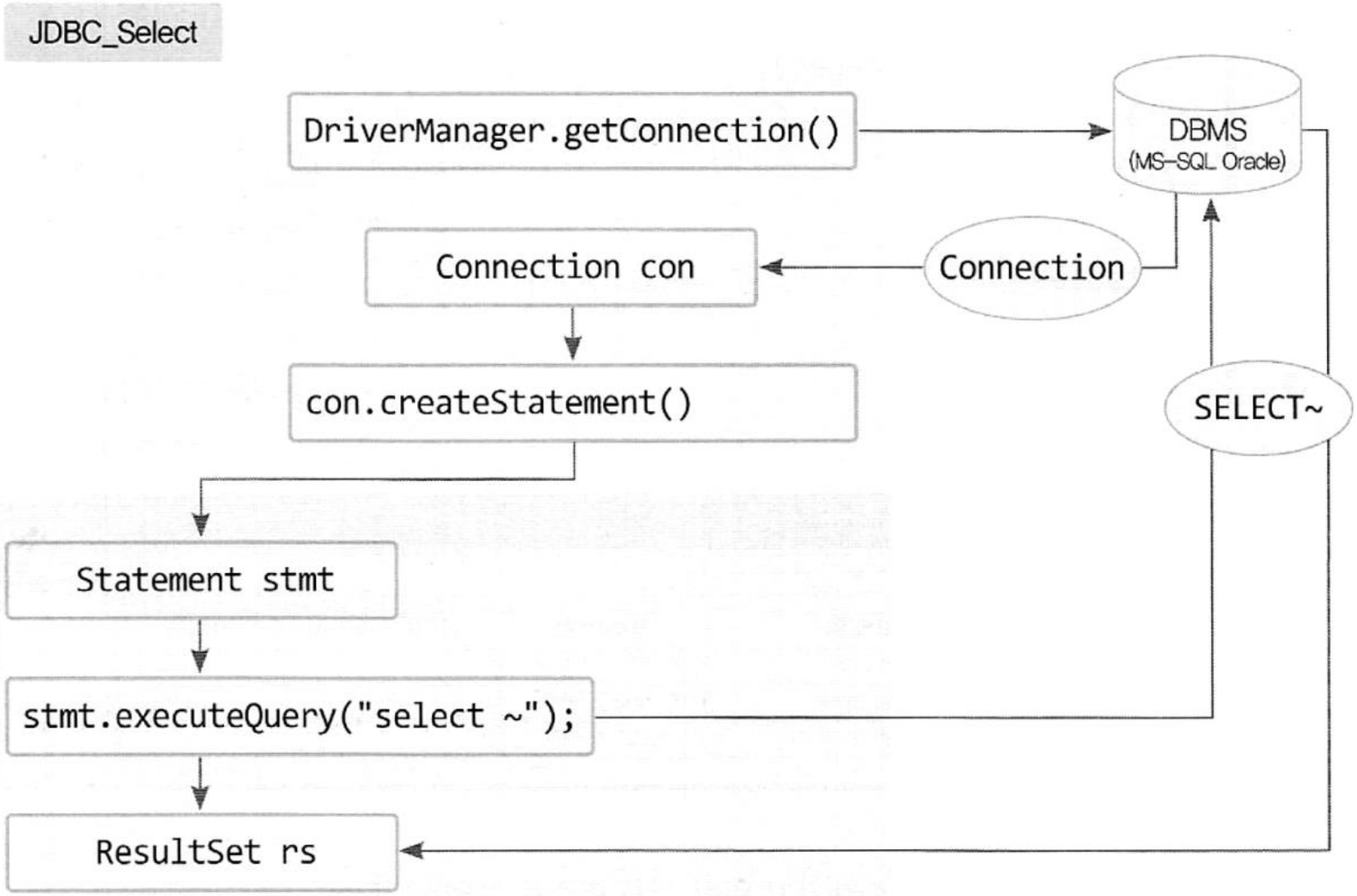
```
<%
   try {
      Class.forName("oracle.jdbc.driver.OracleDriver");
      conn = DriverManager.getConnection(url, uid, pass);
      stmt = conn.createStatement();
      rs = stmt.executeQuery(sql);
      while (rs.next()) {
         out.println("<tr>");
         out.println("<td>" + rs.getString("name") + "</td>");
         out.println("<td>" + rs.getString("userid") + "</td>");
         out.println("<td>" + rs.getString("pwd") + "</td>");
         out.println("<td>" + rs.getString("email") + "</td>");
         out.println("<td>" + rs.getString("phone") + "</td>");
         out.println("<td>" + rs.getInt("admin") + "</td>");
         out.println("</tr>");
      }//while의 끝
   }
```

## ❖ 01_allMember.jsp

```
            catch (Exception e) {
               e.printStackTrace();
            } finally {
               try {
                  if (rs != null)
                     rs.close();
                  if (stmt != null)
                     stmt.close();
                  if (conn != null)
                     conn.close();
               } catch (Exception e) {
                  e.printStackTrace();
               }
            }//finally의 끝
         %>
      </table>
</body>
</html>
```

# JDBC를 이용한 데이터 조작하기

## ❖ SELECT 쿼리 실행



## ❖ 데이터 저장과 PreparedStatement 클래스

- INSERT 쿼리

```
insert into member values('강현승', 'liver', '1234', 'liver@naver.com',
'010-3333-3232', 0)
```

- String 타입으로 쿼리문 준비, Statement의 executeUpdate() 호출

```
String sql="insert into member values('강현승', 'liver', '1234',
'liver@naver.com','010-3333-3232', 0)";
stmt.executeUpdate(sql);
```

## ❖ 데이터 저장과 PreparedStatement 클래스

- ㅇ 저장할 정보가 외부입력으로 결정된다면
  - 변수에 외부 입력 정보를 저장하고 String타입의 쿼리 작성

```
String name   = "강현승";
String userid = "liver";
String pwd    = "1234";
String email  = "moon@nate.com";
String phone  = "010-1111-1111";
String admin  = "0";
String sql    = "insert into member values('" + name  + "', '" + userid
                                              + "', '" + pwd   + "', '" + email
                                              + "', '" + phone + "', "  + admin
                                              + ")";
```

  - 쿼리 작성이 매우 번거로움

## ❖ 데이터 저장과 PreparedStatement 클래스

- PreparedStatement 클래스
  - 쿼리 매개변수 처리 지원
  - Connection의 prepareStatement() 메서드로 생성

```
PreparedStatement pstmt = con.preparedStatement(sql);
```

  - 인자로 SQL문을 전달
    - 매개변수 위치에 ?기호로 서술

```
String sql = "insert into member values(?, ?, ?, ?, ?, ?");
```

## ❖ 데이터 저장과 PreparedStatement 클래스

- PreparedStatement 클래스
  - 매개변수 바인딩

```
set데이터타입(int 순서, 실제 데이터나 변수);
```

```
pstmt.setString(1, name);
pstmt.setString(2, userid);
pstmt.setString(3, pwd);
pstmt.setString(4, email);
pstmt.setString(5, phone);
pstmt.setInt(6, Integer.parseInt(admin));
```

## ❖ 데이터 저장과 PreparedStatement 클래스

- PreparedStatement 클래스
  - 쿼리 실행 : executeUpdate() 메서드 호출

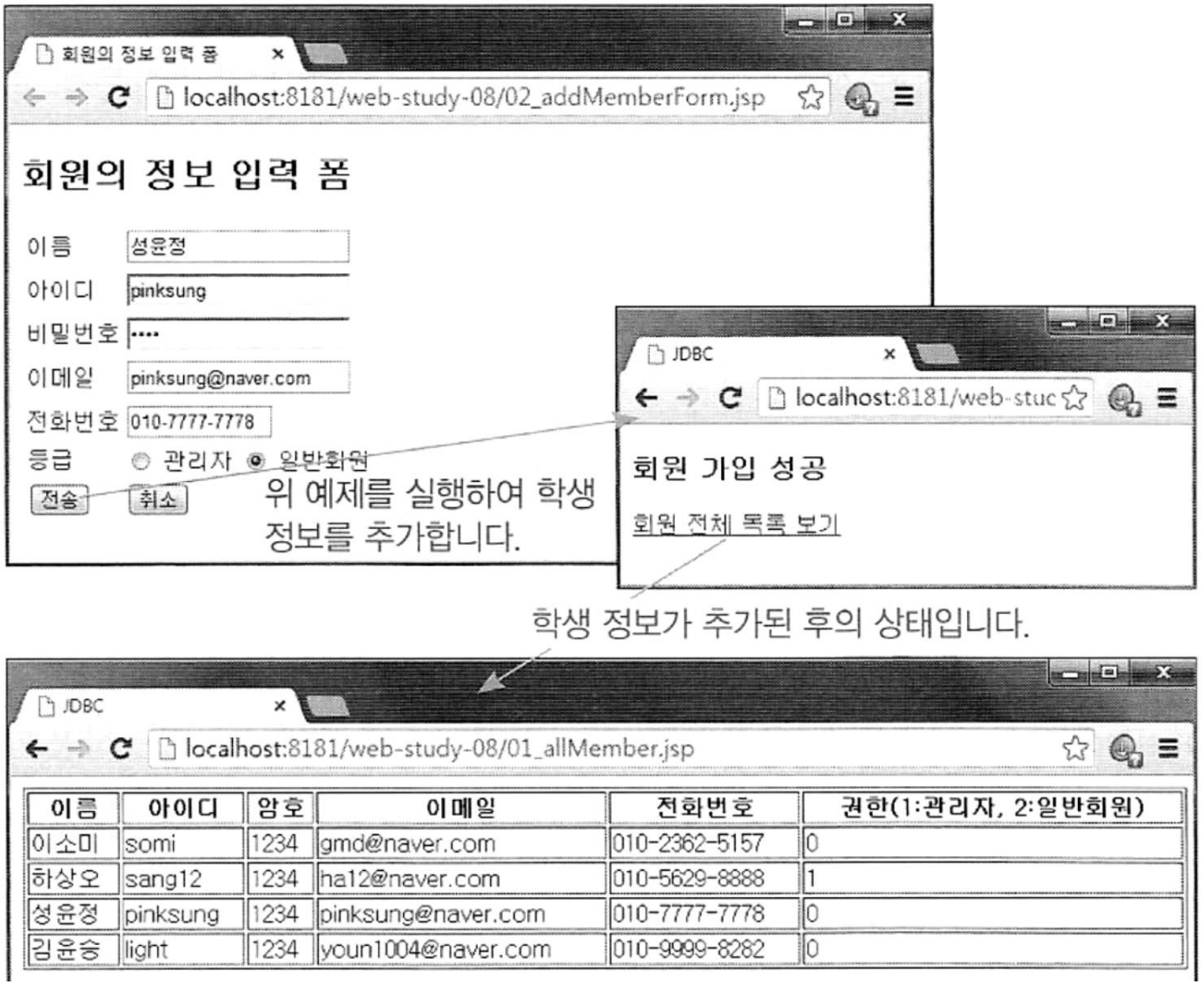
```
pstmt.executeUpdate();
pstmt.close();
```

## ❖ INSERT 문 실행

| 파일 이름 | 설명 |
|---|---|
| 02_addMemberForm.jsp | 이름, 아이디, 비밀번호, 이메일, 전화번호, 등급을 입력 받는 폼이다. [전송] 버튼을 누르면 JSP 파일(addMember.jsp)로 입력된 정보가 전송된다. |
| 02_addMember.jsp | addMemberForm.jsp 문서에서 입력한 정보를 읽어 와서 데이터베이스에 저장한다. |

# JDBC를 이용한 데이터 조작하기

## ❖ INSERT 문 실행




| 이름 | 아이디 | 암호 | 이메일 | 전화번호 | 권한(1:관리자, 2:일반회원) |
|---|---|---|---|---|---|
| 이소미 | somi | 1234 | gmd@naver.com | 010-2362-5157 | 0 |
| 하상오 | sang12 | 1234 | ha12@naver.com | 010-5629-8888 | 1 |
| 성윤정 | pinksung | 1234 | pinksung@naver.com | 010-7777-7778 | 0 |
| 김윤승 | light | 1234 | youn1004@naver.com | 010-9999-8282 | 0 |

## ❖ 02_addMemberForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
:
<body>
    <h2>회원의 정보 입력 폼</h2>
    <form method="post" action="02_addMember.jsp">
        <table>
            <tr>
                <td>이름</td>
                <td><input type="text" name="name" size="20"></td>
            </tr>
            <tr>
                <td>아이디</td>
                <td><input type="text" name="userid" size="20"></td>
            </tr>
            <tr>
                <td>비밀번호</td>
                <td><input type="password" name="pwd" size="20"></td>
            </tr>
```

## ❖ 02_addMemberForm.jsp

```
            <tr>
                <td>이메일</td>
                <td><input type="text" name="email" size="20"></td>
            </tr>
            <tr>
                <td>전화번호</td>
                <td><input type="text" name="phone" size="11"></td>
            </tr>
            <tr>
                <td>등급</td>
                <td>
                    <input type="radio" name="admin" value="1"
                           checked="checked"> 관리자
                    <input type="radio" name="admin"
                           value="0"> 일반회원
                </td>
            </tr>
            <tr>
                <td><input type="submit" value="전송"></td>
                <td><input type="reset" value="취소"></td>
            </tr>
        </table>
    </form>
</body>
</html>
```

## ❖ 02_addMember.jsp

```
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.Connection"%>
<%@page import="java.sql.PreparedStatement"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%!Connection conn = null;
    PreparedStatement pstmt = null;
    String url = "jdbc:oracle:thin:@localhost:1521:XE";
    String uid = "scott";
    String pass = "tiger";
    String sql = "insert into member values(?, ?, ?, ?, ?, ?)";%>
:
<body>
    <%
        request.setCharacterEncoding("UTF-8");
        String name = request.getParameter("name");
        String userid = request.getParameter("userid");
        String pwd = request.getParameter("pwd");
        String email = request.getParameter("email");
        String phone = request.getParameter("phone");
        String admin = request.getParameter("admin");
```

## ❖ 02_addMember.jsp

```
try {
   //(1 단계) JDBC 드라이버 로드
   Class.forName("oracle.jdbc.driver.OracleDriver");

   //(2 단계) 데이터베이스 연결 객체인 Connection 생성
   conn = DriverManager.getConnection(url, uid, pass);

   //(3 단계) Statement 객체 생성하기
   pstmt = conn.prepareStatement(sql);

   //(4 단계) 바인딩 변수를 채운다.
   pstmt.setString(1, name);
   pstmt.setString(2, userid);
   pstmt.setString(3, pwd);
   pstmt.setString(4, email);
   pstmt.setString(5, phone);
   pstmt.setInt(6, Integer.parseInt(admin));

   //(5단계) SQL문을 실행하여 결과 처리
   pstmt.executeUpdate();
}
```

## ❖ 02_addMember.jsp

```
        catch (Exception e) {
            e.printStackTrace();
        } finally {
            //(6단계) 사용한 리소스 해제
            try {
                if (pstmt != null)
                    pstmt.close();
                if (conn != null)
                    conn.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }//finally의 끝
    %>

    <h3>회원 가입 성공</h3>
    <a href="01_allMember.jsp"> 회원 전체 목록 보기 </a>

</body>
</html>
```