**데이터 목록 조회 #2**

2019-11-06

이승진

**학습목표**

조회 조건으로 DB 조회하기

SQL LIKE 연산자

**목차**

[1. 배경지식 2](#_Toc23928804)

[1) LIKE 연산자 2](#_Toc23928805)

[2) PreparedStatement 파라미터 3](#_Toc23928806)

[2. form 태그 구현 4](#_Toc23928807)

[1) form1.jsp 4](#_Toc23928808)

[2) form2.jsp 5](#_Toc23928809)

[3. 이름으로 조회하기 구현 6](#_Toc23928810)

[1) StudentDAO2.java 6](#_Toc23928811)

[2) studentList2.jsp 7](#_Toc23928812)

[3) 분석 9](#_Toc23928813)

[4. 실습 10](#_Toc23928814)

# 배경지식

## LIKE 연산자

SQL의 LIKE 연산자는 문자열 패턴을 비교한다.

|  |
| --- |
| name LIKE '**문자열**%' |

name 필드 값의 시작 부분이 '**문자열**'로 시작하면 true 이다.

|  |
| --- |
| name LIKE '%**문자열**%' |

name 필드 값에 '**문자열**'이 들어있으면 true 이다.

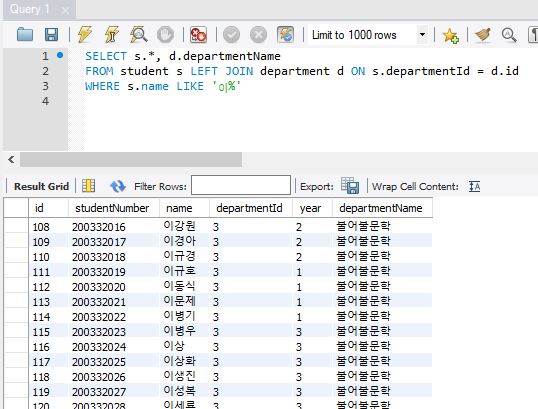
|  |
| --- |
| name LIKE '%**문자열**' |

name 필드 값이 끝부분이 '**문자열**'로 끝나면 true 이다.

### 학생 이름으로 조회

|  |
| --- |
| SELECT s.\*, d.departmentName  FROM student s LEFT JOIN department d ON s.departmentId = d.id  WHERE s.name LIKE '이%' |

학생 이름이 '이'로 시작하는 레코드를 조회하는 SQL



## PreparedStatement 파라미터

|  |
| --- |
| String sql = "SELECT s.\*, d.departmentName " +  "FROM student s LEFT JOIN department d ON s.departmentId = d.id " +  "WHERE s.name LIKE **?**"; |

SQL 명령에서 ? 부분에 주목하자.

SQL 명령을 실행하기 전에, ? 부분에 조회조건 파라미터 값이 삽입되어야 한다.

|  |
| --- |
| String s = name + "%";  statement.**setString**(1, s); |

첫 번째 ? 부분에, 변수 s의 값을 삽입한다.

예를 들어 name 변수 값이 "이" 문자열이었다면, name + "%" 값은 "이%" 이고,

이 값이 삽입된 SQL 명령은 다음과 같다.

|  |
| --- |
| SELECT s.\*, d.departmentName  FROM student s LEFT JOIN department d ON s.departmentId = d.id  WHERE s.name LIKE '이%' |

위 SQL 명령이 DB에서 실행된다.

### 정수 값

|  |
| --- |
| String sql = "SELECT s.\*, d.departmentName " +  "FROM student s LEFT JOIN department d ON s.departmentId = d.id " +  "WHERE s.departmentId = **?**"; |

|  |
| --- |
| statement.**setInt**(1, departmentId); |

첫 번째 ? 부분에, 변수 departmentId의 값을 삽입한다.

예를 들어 departmentId 변수 값이 3 이었다면,

이 값이 삽입된 SQL 명령은 다음과 같다.

|  |
| --- |
| SELECT s.\*, d.departmentName  FROM student s LEFT JOIN department d ON s.departmentId = d.id  WHERE s.departmentId = 3 |

위 SQL 명령이 DB에서 실행된다.

### PreparedStatement 파라미터 값 전달 메소드

void setString(int 파라미터번호, String 파라미터값);

void setInt(int 파라미터번호, int 파라미터값);

void setBoolean(int 파라미터번호, boolean 파라미터값);

void setDate(int 파라미터번호, Date 파라미터값);

void setDouble(int 파라미터번호, double 파라미터값);

void setTimestamp(int 파라미터번호, Timestamp 파라미터값);

# form 태그 구현

## form1.jsp

WebContent/jdbc1/form1.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%  String srchText = request.getParameter("srchText");  %>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  </head>  <body>  <form>  <label>이름</label>  <input type="text" name="srchText" />  <button type="submit">조회</button>  </form>  <hr />  <div>  srchText 값: <%= srchText %>  </div>  </body>  </html> |

String srchText = request.getParameter("srchText");

웹브라우저에서 서버로 전송된 데이터들을 request parameter 라고 한다.

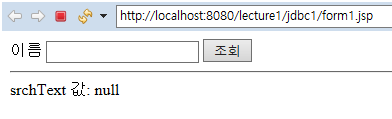
request parameter 중에서, 데이터 이름이 srchText인 항목의 값을 꺼내서 srchText 변수에 대입한다.

만약 이 이름의 request parameter가 없다면, null 값이 리턴된다.

form1.jsp를 처음 실행했을 때는, request parameter가 전혀 없기 때문에,

request.getParameter("srchText") 메소드 호출은 null을 리턴한다.

그래서 아래와 같이 출력된다.



<form>

<label>이름</label>

<input type="text" name="srchText" />

<button type="submit">조회</button>

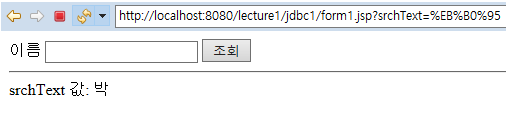
</form>

조회 버튼을 클릭하면, 웹브라우저의 주소칸의 현재 URL이 서버에 다시 요청(request)된다.

이때 "srchText" 이름의 텍스트박스에 입력한 값이, 요청(request)에 담겨서 서버에 전송된다.

이 데이터를 request parameter 라고 한다.

텍스트 박스에 "박"을 입력하고 조회 버튼을 클릭해 보자.



"srchText" 이름의 request parameter 값이 출력되었다.

## form2.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%  String srchText = request.getParameter("srchText");  %>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  </head>  <body>  <form>  <label>이름</label>  <input type="text" name="srchText" value="<%= srchText %>"/>  <button type="submit">조회</button>  </form>  <hr />  <div>  srchText 값: <%= srchText %>  </div>  </body>  </html> |

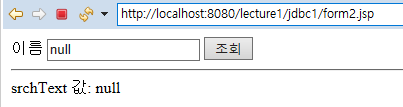
<input type="text" name="srchText" value="<%= srchText %>"/>

텍스트 박스가 웹브라우저에 그려질 때, srchText 변수 값이 채우져서 그려진다.

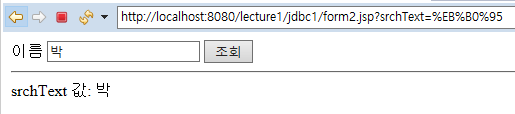
form2.jsp를 처음 실행했을 때는, request parameter가 전혀 없기 때문에,

request.getParameter("srchText") 메소드 호출은 null을 리턴한다.

그래서 텍스트박스에도 null이 채워진다.



텍스트 박스에 "박"을 입력하고 조회 버튼을 클릭해 보자.



"srchText" 이름의 request parameter 값이 텍스트박스에 채워졌다.

URL query string 부분의 ?srchText=%EB%B0%95 부분에 주목하자.

서버에 전달된 request parameter 이름과 값이, URL의 query string으로 표현되었다.

URL에 한글이 포함될 수 없으므로, 한글은 16진수 형태로 변환되어 (url encoding 되어) 표현되었다.

# 이름으로 조회하기 구현

## StudentDAO.java

src/lecture1/jdbc1/StudentDAO.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57 | package lecture1.jdbc1;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.util.ArrayList;  import java.util.List;  import lecture1.DB;  public class StudentDAO {  public static List<Student> findAll() throws Exception {  String sql = "SELECT s.\*, d.departmentName " +  "FROM student s LEFT JOIN department d ON s.departmentId = d.id";  try (Connection connection = DB.getConnection("student1");  PreparedStatement statement = connection.prepareStatement(sql);  ResultSet resultSet = statement.executeQuery()) {  ArrayList<Student> list = new ArrayList<Student>();  while (resultSet.next()) {  Student student = new Student();  student.setId(resultSet.getInt("id"));  student.setStudentNumber(resultSet.getString("studentNumber"));  student.setName(resultSet.getString("name"));  student.setDepartmentId(resultSet.getInt("departmentId"));  student.setYear(resultSet.getInt("year"));  student.setDepartmentName(resultSet.getString("departmentName"));  list.add(student);  }  return list;  }  }  public static List<Student> findByName(String name) throws Exception {  String sql = "SELECT s.\*, d.departmentName " +  "FROM student s LEFT JOIN department d ON s.departmentId = d.id " +  "WHERE s.name LIKE ?";  try (Connection connection = DB.getConnection("student1");  PreparedStatement statement = connection.prepareStatement(sql)) {  statement.setString(1, name + "%");  try (ResultSet resultSet = statement.executeQuery()) {  ArrayList<Student> list = new ArrayList<Student>();  while (resultSet.next()) {  Student student = new Student();  student.setId(resultSet.getInt("id"));  student.setStudentNumber(resultSet.getString("studentNumber"));  student.setName(resultSet.getString("name"));  student.setDepartmentId(resultSet.getInt("departmentId"));  student.setYear(resultSet.getInt("year"));  student.setDepartmentName(resultSet.getString("departmentName"));  list.add(student);  }  return list;  }  }  }  } |

줄21~28 부분과 줄44~51 부분이 동일하다.

동일한 소스코드가 여기 저기 반복된다면, 동일한 부분을 메소드로 추출하여 구현하는 것이 바람직하다.

## studentList2.jsp

WebContent/jdbc2/studentList2.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ page import="java.util.List, lecture1.jdbc1.\*" %>  <%  String srchText = request.getParameter("srchText");  if (srchText == null) srchText = "";  List<Student> list = StudentDAO.findByName(srchText);  %>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>  <style>  thead th { background-color: #eee; }  table.table { width: 700px; margin-top: 10px; }  </style>  </head>  <body>  <div class="container">  <h1>학생목록</h1>  <form class="form-inline">  <div class="form-group">  <label>이름</label>  <input type="text" class="form-control" name="srchText" value="<%= srchText %>"  placeholder="검색조건" />  </div>  <button type="submit" class="btn btn-primary">조회</button>  </form>  <table class="table table-bordered table-condensed">  <thead>  <tr>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>학년</th>  </tr>  </thead>  <tbody>  <% for (Student student : list) { %>  <tr>  <td><%= student.getStudentNumber() %></td>  <td><%= student.getName() %></td>  <td><%= student.getDepartmentName() %></td>  <td><%= student.getYear() %></td>  </tr>  <% } %>  </tbody>  </table>  </div>  </body>  </html> |

if (srchText == null) srchText = "";

"srchText" 이름의 request parameter가 없다면, srchText 변수에 "" 을 대입한다.

List<Student> list = StudentDAO2.findByName(srchText);

만약 "srchText" 이름의 request parameter 값이 "이" 였다면,

srchText 변수 값이 "이" 이므로,

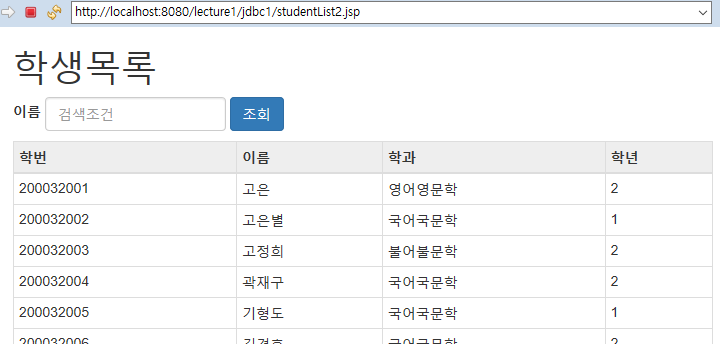
WHERE s.name LIKE "이%" 조회조건으로 조회하게 된다.

만약 "srchText" 이름의 request parameter 항목이 었다면,

srchText 변수 값이 "" 이므로,

WHERE s.name LIKE "%" 조회조건으로 조회하게 된다.

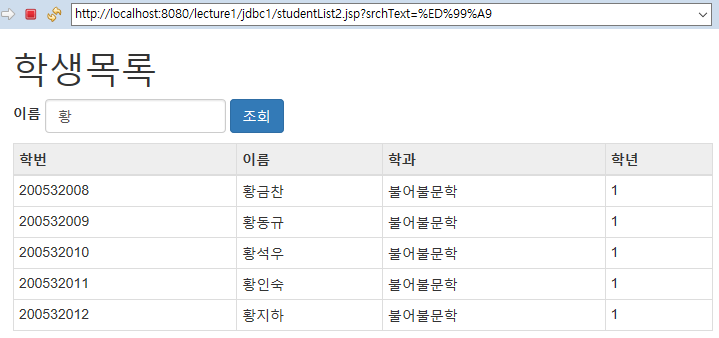
## 분석



처음 실행하면 전체 학생 목록이 조회된다.

그 이유는?

검색 조건 텍스트박스에 "황"을 입력하고 조회 버튼을 클릭하자.



황씨만 조회되는 이유는?

검색 조건 텍스트박스에 "황"이 채워져 있는 이유는?

# 실습

User 테이블에 대해서

이름으로 조회하기 페이지를 구현하라