**페이지 단위 조회**

2017-04-06

이승진

**학습목표**

페이지 단위로 DB 레코드 조회하기

JSP tag file 구현

**목차**

[1. 배경지식 2](#_Toc514833658)

[1) query string과 request parameter 2](#_Toc514833659)

[2) 페이지 단위 조회 SQL 명령 - MySQL 문법 3](#_Toc514833660)

[3) 페이지 단위 조회 SQL 명령 - SQL Server 문법 5](#_Toc514833661)

[4) JSP 태그 파일 (JSP tag file) 7](#_Toc514833662)

[5) JSP 태그 파일 생성 7](#_Toc514833663)

[6) JSP 태그 파일 사용 8](#_Toc514833664)

[2. 데이터 조회 Java 코드 10](#_Toc514833665)

[1) Student.java 10](#_Toc514833666)

[2) StudentDAO.java 11](#_Toc514833667)

[3. JSP 파일 구현 13](#_Toc514833668)

[1) jdbc2/studentList1.jsp 13](#_Toc514833669)

[2) jdbc2/studentList2.jsp 15](#_Toc514833670)

[3) jdbc2/studentList3.jsp 17](#_Toc514833671)

[4. 연습문제 20](#_Toc514833672)

[5. 검색 & 페이지 단위 조회 21](#_Toc514833673)

[1) StudentDAO2.java 21](#_Toc514833674)

[2) studentList4.jsp 22](#_Toc514833675)

[3) 실행 과정 설명 23](#_Toc514833676)

[4) 실행 과정 설명 26](#_Toc514833677)

# 배경지식

## query string과 request parameter

웹브라우저가 서버에 URL을 요청(http request) 할 때,

요청되는 URL의 query string 부분이 request parameter 데이터에 포함된다.

서버에서 이 request parameter 데이터를 꺼낼 수 있다.

예를 들어 요청되는 URL 이 다음과 같다고 하자.

|  |
| --- |
| http://localhost:8080/lecture1/jdbc2/studentList3.jsp?pg=5 |

위 URL의 query string인 pg=5 부분이 request parameter 데이터에 포함되어 서버에 전달된다.

서버에서 위의 request parameter 데이터를 아래와 같이 꺼낼 수 있다.

|  |
| --- |
| String s = request.getParameter("pg"); |

requst.getParameter(...) 메소드의 리턴 값은 언제나 String 객체이기 때문에

만약 꺼낸 값이 정수이어야 한다면 다음과 같이 변환해야 한다.

|  |
| --- |
| String s = request.getParameter("pg");  int currentPage = Integer.parseInt(s); |

request parameter 값이 실수이라면 다음과 같이 변환한다.

|  |
| --- |
| String s = request.getParameter("weight");  double weight = Double.parseDouble(s); |

request parameter 값이 2017-04-06 형태의 날짜라면 다음과 같이 변환한다.

|  |
| --- |
| SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");  String s = request.getParameter("birthDate");  Date birthDate = format.parse(s); |

## 페이지 단위 조회 SQL 명령 - MySQL 문법

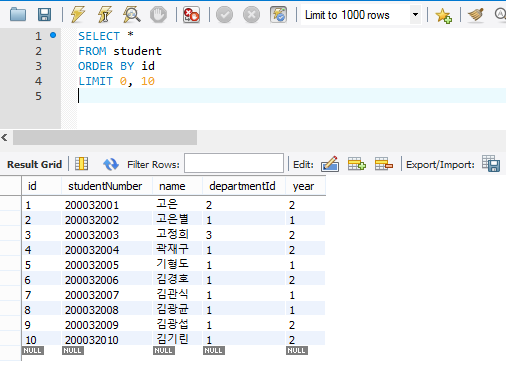
|  |
| --- |
| SELECT \*  FROM student  ORDER BY id |

위 SQL 명령은 student 테이블의 전체 레코드를 id 필드 순서로 조회한다.

### LIMIT 구문

|  |
| --- |
| SELECT \*  FROM student  ORDER BY id  LIMIT 0, 10 |

위 SQL 명령은 student 테이블에서 id 필드 순서로 0 번째 레코드 부터 10개의 레코드만 조회한다.



|  |
| --- |
| SELECT \*  FROM student  ORDER BY id  LIMIT 20, 10 |

위 SQL 명령은 student 테이블에서 id 필드 순서로 20 번째 레코드 부터 10개의 레코드만 조회한다.

|  |
| --- |
| SELECT \*  FROM 테이블명  ORDER BY 필드  LIMIT 첫째\_레코드\_번호, 조회할\_레코드\_수 |

MySQL에서 레코드 번호는 0 번 부터 시작한다.

한 페이지에 15 개 레코드를 보여 준다고 할 때, 3 번째 페이지의 레코드를 조회하려면,

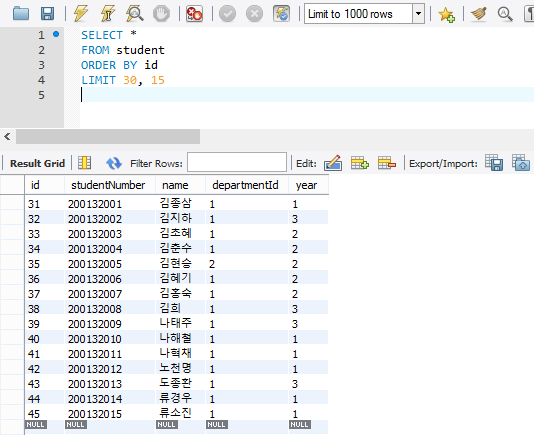
int pageSize = 15;

int currentPage = 3;

int 첫째\_레코드\_번호 = (currentPage - 1) \* pageSize; /\* (3 - 1) \* 15 = 30 \*/

int 조회할\_레코드\_수 = pageSize; /\* 15 \*/

|  |
| --- |
| SELECT \*  FROM 테이블명  ORDER BY 필드  LIMIT 30, 15 |



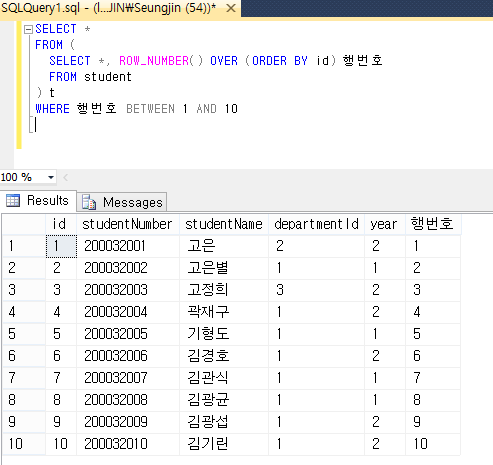
## 페이지 단위 조회 SQL 명령 - Microsoft SQL Server 문법

|  |
| --- |
| SELECT \*  FROM (  SELECT \*, ROW\_NUMBER() OVER (ORDER BY id) 행번호  FROM student  ) t  WHERE 행번호 BETWEEN 1 AND 10 |

위 명령은 student 테이블에서 id 순서로 1 번 레코드에서 10 번 레코드까지

10개의 레코드를 조회한다.

레코드 번호는 1 부터 시작한다.



|  |
| --- |
| SELECT \*  FROM (  SELECT \*, ROW\_NUMBER() OVER (ORDER BY 필드) 행번호  FROM 테이블  ) t  WHERE 행번호 BETWEEN 첫째\_레코드\_번호 AND 마지막\_레코드\_번호 |

SQL Server에서 레코드 번호는 1 부터 시작한다.

한 페이지에 15 개 레코드를 보여 준다고 할 때, 3 번째 페이지의 레코드를 조회하려면,

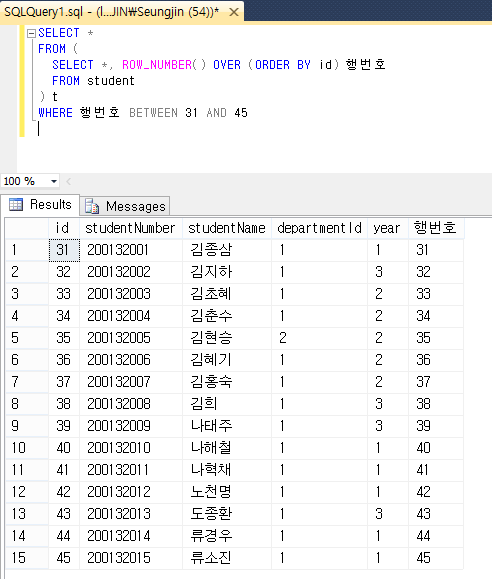
int pageSize = 15;

int currentPage = 3;

int 첫째\_레코드\_번호 = (currentPage - 1) \* pageSize + 1; /\* (3 - 1) \* 15 = 31 \*/

int 마지막\_레코드\_번호 = 첫째\_레코드\_번호 + pageSize; /\* 31 + 15 - 1 = 45 \*/

|  |
| --- |
| SELECT \*  FROM (  SELECT \*, ROW\_NUMBER() OVER (ORDER BY id) 행번호  FROM student  ) t  WHERE 행번호 BETWEEN 31 AND 45 |



## JSP 태그 파일 (JSP tag file)

여러 JSP 페이지들에서 동일한 기능을 반복적으로 구현해야 한다면,

그 부분을 JSP 태그 파일 형태로 구현하면, 여러 JSP 페이지에서 간단하게 호출할 수 있다.

예를 들어서, 현재 날짜와 시을 출력하는 기능을 여러 JSP 페이지들에 반복적으로 구현해야 한다면,

현재 날짜와 시각을 JSP 태그 파일 형태로 구현하자.

그리고 그 태그 파일을 JSP 페이지에서 호출하자.

JSP 태그 파일의 확장자는 \*.tag 이어야 한다.

JSP 태그 파일은 WebContent/WEB-INF 폴더 아래에 위치해야 한다.

## JSP 태그 파일 생성

### WebContent/WEB-INF/tags 폴더 생성

JSP 태그 파일들을 모아 놓을 폴더를 생성하자.

WebContent/WEB-INF 폴더 아래에 tags 폴더를 생성하자.

WebContent/WEB-INF 폴더를 마우스 오른쪽 버튼으로 클릭하고

메뉴에서 New - Folder

Folder Name: tags

### WebContent/WEB-INF/tags/dateTime.tag 파일 생성

WebContent/WEB-INF/tags 폴더를 마우스 오른쪽 버튼으로 클릭하고

메뉴에서 New - File

File Name: dateTime.tag

WebContent/WEB-INF/tags/dateTime.tag

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | <%@ tag description="pagination" pageEncoding="UTF-8"%>  <%@ tag import="java.text.SimpleDateFormat, java.util.Date" %>  <%@ attribute name="format" type="java.lang.String" required="true" %>  <%@ attribute name="date" type="java.util.Date" required="false" %>  <%  String format = (String)jspContext.getAttribute("format");  Date date = (Date)jspContext.getAttribute("date");  if (date == null) date = new Date();  SimpleDateFormat simpleDateFormat = new SimpleDateFormat(format);  %>  <span>  <%= simpleDateFormat.format(date) %>  </span> |

(줄2) 본문에서 SimpleDateFormat 클래스와 Date 클래스를 사용하기 때문에, 이 import 선언이 필요하다.

(줄3) JSP 페이지에서 dateTime.tag 태그 파일을 호출할 때, format 파라미터를 사용할 수 있다.

format 파라미터는 필수 파라미터이다. (required=true)

(줄4) JSP 페이지에서 dateTime.tag 태그 파일을 호출할 때, date 파라미터를 사용할 수 있다.

date 파라미터는 필수 파라미터가 아니다. (required=false)

(줄6) format 파라미터 값을 꺼낸다.

(줄7) date 파라미터 값을 꺼낸다.

## JSP 태그 파일 사용

### WebContent/jdbc3/testTagFile.jsp 생성

|  |  |
| --- | --- |
| 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 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ page import="java.util.Date" %>  <%@ taglib tagdir="/WEB-INF/tags" prefix="my" %>  <!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>  </head>  <body>  <div class="container">  <h1>JSP 태그 파일 테스트</h1>  <hr />  <p>  <my:dateTime format="yyyy-MM-dd" />  </p>  <p>  <my:dateTime format="yyyy-MM-dd HH:mm" />  </p>  <p>  <my:dateTime format="yyyy-MM-dd HH:mm:ss" date="<%= new Date() %>" />  </p>  </div>  </body>  </html> |

(줄3)

|  |
| --- |
| <%@ taglib tagdir="/WEB-INF/tags" prefix="my" %> |

WebContent/WEB-INF/tags/\*.tag 태그 파일들을 사용하려면,

JSP 페이지 선두에 이 선언이 필요하다.

(줄21)

|  |
| --- |
| <my:dateTime format="yyyy-MM-dd" /> |

WebContent/WEB-INF/tags/dateTime.tag 파일을 호출한다.

format 파라미터 값으로 "yyyy-MM-dd" 문자열을 전달한다.

date 파라미터 값은 전달하지 않는다.

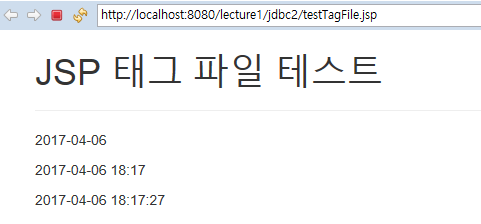
(줄29)

|  |
| --- |
| <my:dateTime format="yyyy-MM-dd HH:mm:ss" date="<%= new Date() %>" /> |

WebContent/WEB-INF/tags/dateTime.tag 파일을 호출한다.

format 파라미터 값으로 "yyyy-MM-dd HH:mm:ss" 문자열을 전달한다.

date 파라미터 값으로 new Date() 표현식 값을 전달한다.



# 데이터 조회 Java 코드

## Student.java

### src/lecture1/jdbc2/Student.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  58  59 | package lecture1.jdbc2;  public class Student {  int id;  String studentNumber;  String name;  int departmentId;  int year;  String departmentName;  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getStudentNumber() {  return studentNumber;  }  public void setStudentNumber(String studentNumber) {  this.studentNumber = studentNumber;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public int getDepartmentId() {  return departmentId;  }  public void setDepartmentId(int departmentId) {  this.departmentId = departmentId;  }  public int getYear() {  return year;  }  public void setYear(int year) {  this.year = year;  }  public String getDepartmentName() {  return departmentName;  }  public void setDepartmentName(String departmentName) {  this.departmentName = departmentName;  }  } |

Student 테이블에서 조회한 데이터를 채워서 전달하기 위한 객체이다. DTO 객체

|  |  |
| --- | --- |
| package | lecture1.jdbc2 |
| class name | Student |

## StudentDAO.java

### lecture1/jdbc2/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 | package lecture1.jdbc2;  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(int currentPage, int pageSize) throws Exception {  String sql = "SELECT s.\*, d.departmentName" +  " FROM student s LEFT JOIN department d ON s.departmentId = d.id" +  " LIMIT ?, ?";  try (Connection connection = DB.getConnection("student1");  PreparedStatement statement = connection.prepareStatement(sql)) {  statement.setInt(1, (currentPage - 1) \* pageSize);  statement.setInt(2, pageSize);  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;  }  }  }  public static int count() throws Exception {  String sql = "SELECT COUNT(\*) FROM student";  try (Connection connection = DB.getConnection("student1");  PreparedStatement statement = connection.prepareStatement(sql);  ResultSet resultSet = statement.executeQuery()) {  if (resultSet.next())  return resultSet.getInt(1);  }  return 0;  }  } |

|  |  |
| --- | --- |
| package | lecture1.jdbc2 |
| class name | StudentDAO |

(줄12)

|  |
| --- |
| public static List<Student> findAll(int currentPage, int pageSize) |

Student 테이블의 레코드를 페이지 단위로 조회해서 리턴하는 메소드이다.

currentPage : 조회할 페이지 번호 (페이지 번호는 1 부터 시작한다)

pageSize : 한 페이지의 레코드 수

(줄37)

|  |
| --- |
| public static int count() |

Student 테이블의 전체 레코드 수를 조회해서 리턴하는 메소드이다.

(줄13~15)

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

실행할 SQL 명령이다.

이 SQL 명령을 실행하기 전에, 물음표(?) 부분에 파라미터 값이 삽입되어야 한다.

(줄18)

|  |
| --- |
| statement.setInt(1, (currentPage - 1) \* pageSize); |

실행할 SQL 명령의 첫 번째 물음표(?) 위치에 (currentPage - 1) \* pageSize 표현식의 값이 삽입된다.

(currentPage - 1) \* pageSize 표현식은 조회할 첫째 레코드 번호를 계산한다.

(줄19)

|  |
| --- |
| statement.setInt(2, pageSize); |

실행할 SQL 명령의 두 번째 물음표(?) 위치에 pageSize 값이 삽입된다.

pageSize 값은 조회할 레코드 수 이다.

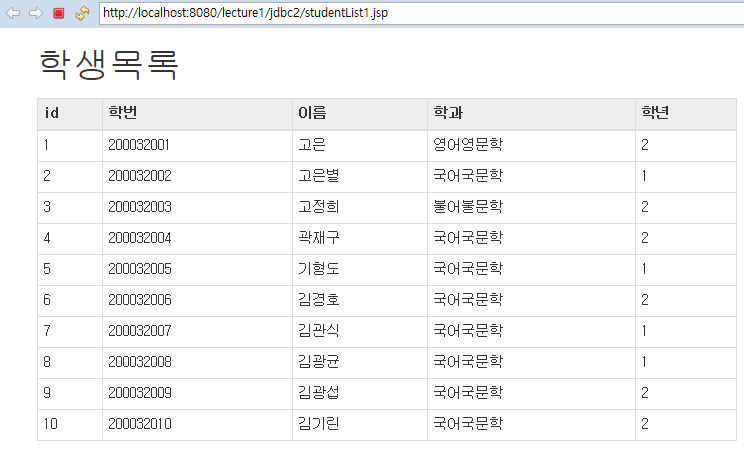
# JSP 파일 구현

## jdbc2/studentList1.jsp

### WebContent/jdbc2/studentList1.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 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ page import="java.util.List, lecture1.jdbc2.\*" %>  <%  int currentPage = 1;  int pageSize = 10;  List<Student> list = StudentDAO.findAll(currentPage, pageSize);  %>  <!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>  body { font-family: 굴림체; }  thead th { background-color: #eee; }  table.table { width: 700px; }  </style>  </head>  <body>  <div class="container">  <h1>학생목록</h1>  <table class="table table-bordered table-condensed">  <thead>  <tr>  <th>id</th>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>학년</th>  </tr>  </thead>  <tbody>  <% for (Student student : list) { %>  <tr>  <td><%= student.getId() %></td>  <td><%= student.getStudentNumber() %></td>  <td><%= student.getName() %></td>  <td><%= student.getDepartmentName() %></td>  <td><%= student.getYear() %></td>  </tr>  <% } %>  </tbody>  </table>  </div>  </body>  </html> |

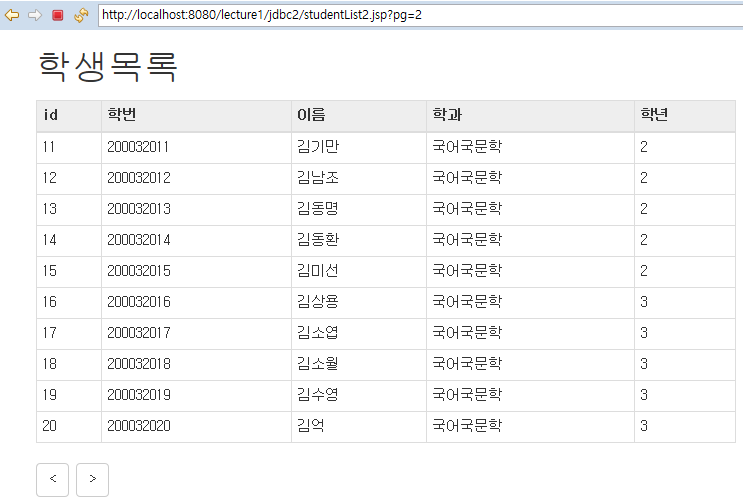
위 예제는 첫 번째 페이지의 10개 레코드만 조회한다.



## jdbc2/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  60  61  62  63  64  65 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ page import="java.util.\*, lecture1.jdbc2.\*" %>  <%  int currentPage = 1;  int pageSize = 10;  String pg = request.getParameter("pg");  if (pg != null) currentPage = Integer.parseInt(pg);  List<Student> list = StudentDAO.findAll(currentPage, pageSize);  int recordCount = StudentDAO.count();  int pageCount = (int)Math.ceil((double)recordCount / pageSize);  %>  <!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>  body { font-family: 굴림체; }  thead th { background-color: #eee; }  table.table { width: 700px; }  </style>  </head>  <body>  <div class="container">  <h1>학생목록</h1>  <table class="table table-bordered table-condensed">  <thead>  <tr>  <th>id</th>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>학년</th>  </tr>  </thead>  <tbody>  <% for (Student student : list) { %>  <tr>  <td><%= student.getId() %></td>  <td><%= student.getStudentNumber() %></td>  <td><%= student.getName() %></td>  <td><%= student.getDepartmentName() %></td>  <td><%= student.getYear() %></td>  </tr>  <% } %>  </tbody>  </table>  <% if (currentPage > 1) { %>  <a class="btn btn-default" href="studentList2.jsp?pg=<%= currentPage-1 %>"> &lt; </a>  <% } %>  <% if (currentPage < pageCount) { %>  <a class="btn btn-default" href="studentList2.jsp?pg=<%= currentPage+1 %>"> &gt; </a>  <% } %>  </div>  </body>  </html> |



### 실행 과정 분석

studentList2.jsp 를 실행하자.

(줄56~61) 부분이 어떻게 출력되었는 웹브라우저 소스보기에서 확인하자.

위 화면 아래의 > 버튼을 클릭해보자.

(줄56~61) 부분이 어떻게 출력되었는 웹브라우저 소스보기에서 확인하자.

버튼을 클릭했을 때, 웹브라우저와 웹서버 사이에서 어떤 요청과 응답이 오고 갔는가?

그 자세한 실행과정을 설명하라.

## jdbc2/studentList3.jsp

### WebContent/WEB-INF/tags/pagination.tag

|  |  |
| --- | --- |
| 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  60  61  62 | <%@ tag description="pagination" pageEncoding="UTF-8"%>  <%@ tag import="java.util.ArrayList" %>  <%@ attribute name="recordCount" type="java.lang.Integer" required="true" %>  <%@ attribute name="pageSize" type="java.lang.Integer" required="true" %>  <%@ attribute name="queryStringName" type="java.lang.String" required="true" %>  <%!  private class Page {  int page;  String label;    Page(int page, String label) {  this.page = page;  this.label = label;  }  }  %>  <%  int recordCount = (Integer)jspContext.getAttribute("recordCount");  int pageSize = (Integer)jspContext.getAttribute("pageSize");  String name = (String)jspContext.getAttribute("queryStringName");  int currentPage = 1;  if (request.getParameter(name) != null)  currentPage = Integer.parseInt(request.getParameter(name));  int pageCount = recordCount / pageSize;  if (pageCount \* pageSize < recordCount) ++pageCount;  String queryString = request.getQueryString();  if (queryString == null)  queryString = name + "=@@@";  else if (queryString.matches(".\*" + name + "=[0-9]+.\*"))  queryString = queryString.replaceAll(name + "=[0-9]+", name + "=@@@");  else  queryString = queryString + "&" + name + "=@@@";  Object temp = request.getAttribute("javax.servlet.forward.request\_uri");  String uri = (temp != null) ? temp.toString() : request.getRequestURI();  String url = uri + "?" + queryString;  if (currentPage > pageCount) currentPage = pageCount;  int base = ((currentPage - 1) / 10) \* 10;  ArrayList<Page> pages = new ArrayList<Page>();  if (base > 0) pages.add(new Page(base, "&lt;"));  for (int i = 1; i <= 10; ++i) {  int n = base + i;  if (n > pageCount) break;  pages.add(new Page(n, String.valueOf(n)));  }  int n = base + 11;  if (n <= pageCount)  pages.add(new Page(n, "&gt;"));  %>  <ul class="pagination">  <% for (Page p : pages) { %>  <li class='<%= p.page == currentPage ? "active" : "" %>'>  <a href='<%= url.replace("@@@", String.valueOf(p.page)) %>'><%= p.label %></a>  </li>  <% } %>  </ul> |

pagination.tag 소스를 지금 당장 이해할 필요는 없다.

### WebContent/jdbc2/studentList3.jsp

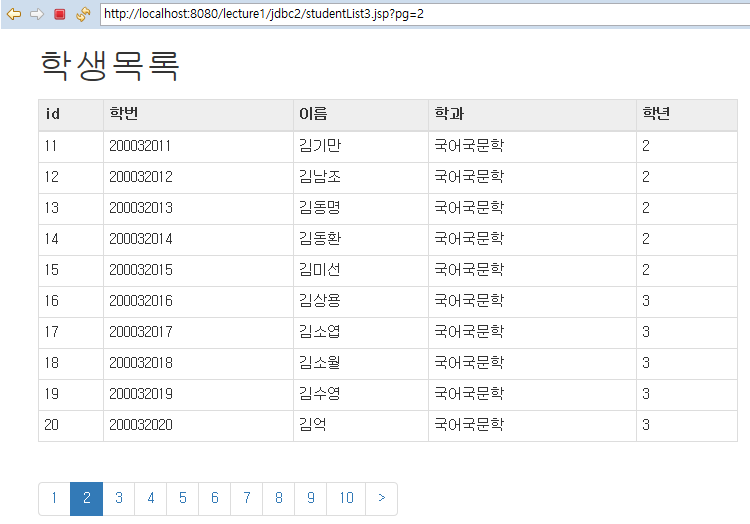
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ page import="java.util.\*, lecture1.jdbc2.\*" %>  <%@ taglib tagdir="/WEB-INF/tags" prefix="my" %>  <%  int currentPage = 1;  int pageSize = 10;  String pg = request.getParameter("pg");  if (pg != null) currentPage = Integer.parseInt(pg);  List<Student> list = StudentDAO.findAll(currentPage, pageSize);  int recordCount = StudentDAO.count();  %>  <!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>  body { font-family: 굴림체; }  thead th { background-color: #eee; }  table.table { width: 700px; }  </style>  </head>  <body>  <div class="container">  <h1>학생목록</h1>  <table class="table table-bordered table-condensed">  <thead>  <tr>  <th>id</th>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>학년</th>  </tr>  </thead>  <tbody>  <% for (Student student : list) { %>  <tr>  <td><%= student.getId() %></td>  <td><%= student.getStudentNumber() %></td>  <td><%= student.getName() %></td>  <td><%= student.getDepartmentName() %></td>  <td><%= student.getYear() %></td>  </tr>  <% } %>  </tbody>  </table>  <my:pagination pageSize="<%= pageSize %>" recordCount="<%= recordCount %>" queryStringName="pg" />  </div>  </body>  </html> |

(줄8) (줄56) 에서 "pg" 문자열은 현재 페이지 번호 값에 해당하는

request parameter 데이터의 이름이자 querstr string 이름이다.

지금은 pagination.tag 의 사용 방법만 익혀두자.

앞으로 페이지 단위 조회 화면을 구현할 때, (줄3)의 선언과 (줄56) 태그만 파일에 추가하면 된다.



### 실행 과정 분석

studentList3.jsp 를 실행하자.

(줄56) 부분이 어떻게 출력되었는 웹브라우저 소스보기에서 확인하자.

페이지 번호를 아무 것이나 클릭해보자.

(줄56) 부분이 어떻게 출력되었는 웹브라우저 소스보기에서 확인하자.

페이지 번호를 클릭했을 때, 웹브라우저와 웹서버 사이에서 어떤 요청과 응답이 오고 갔는가?

그 자세한 실행과정을 설명하라.

# 연습문제

User 테이블의 레코드 목록을 페이지 단위로 조회하는 기능을 구현하라.

# 검색 & 페이지 단위 조회

## StudentDAO2.java

### src/lecture1/jdbc2/StudentDAO2.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 | package lecture1.jdbc2;  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 StudentDAO2 {  public static List<Student> findByName(String name, int currentPage, int pageSize) throws Exception {  String sql = "SELECT s.\*, d.departmentName" +  " FROM student s LEFT JOIN department d ON s.departmentId = d.id" +  " WHERE name LIKE ?" +  " LIMIT ?, ?";  try (Connection connection = DB.getConnection("student1");  PreparedStatement statement = connection.prepareStatement(sql)) {  statement.setString(1, name + "%");  statement.setInt(2, (currentPage - 1) \* pageSize);  statement.setInt(3, pageSize);  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;  }  }  }  public static int count(String name) throws Exception {  String sql = "SELECT COUNT(\*) FROM student WHERE name LIKE ?";  try (Connection connection = DB.getConnection("student1");  PreparedStatement statement = connection.prepareStatement(sql)) {  statement.setString(1, name + "%");  try (ResultSet resultSet = statement.executeQuery()) {  if (resultSet.next())  return resultSet.getInt(1);  }  }  return 0;  }  } |

### findByName 메소드

Student 테이블에서, 이름으로 조회하고, 페이지 단위로 조회하여, 학생 목록을 리턴한다.

### count 메소드

Student 테이블에서, 이름으로 조회하여, 레코드 수를 리턴한다.

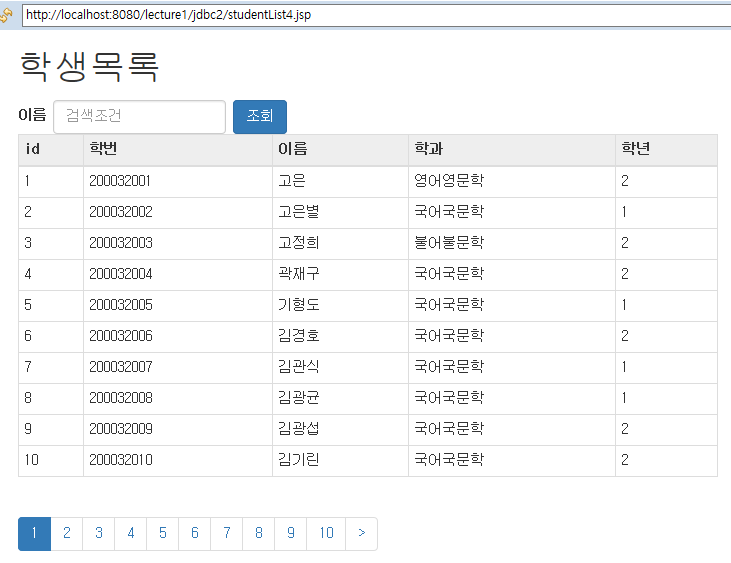
이름으로 조회한 결과의 총 페이지 수를 계산하려면, 조회 결과 레코드 수가 필요하다.

## studentList4.jsp

### WebContent/jdbc2/studentList4.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  60  61  62  63  64  65  66  67  68  69  70  71  72 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ page import="java.util.\*, lecture1.jdbc2.\*" %>  <%@ taglib tagdir="/WEB-INF/tags" prefix="my" %>  <%  int currentPage = 1;  int pageSize = 10;  String pg = request.getParameter("pg");  if (pg != null) currentPage = Integer.parseInt(pg);  String srchText = request.getParameter("srchText");  if (srchText == null) srchText = "";  List<Student> list = StudentDAO2.findByName(srchText, currentPage, pageSize);  int recordCount = StudentDAO2.count(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>  body { font-family: 굴림체; }  thead th { background-color: #eee; }  table.table { width: 700px; }  </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>id</th>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>학년</th>  </tr>  </thead>  <tbody>  <% for (Student student : list) { %>  <tr>  <td><%= student.getId() %></td>  <td><%= student.getStudentNumber() %></td>  <td><%= student.getName() %></td>  <td><%= student.getDepartmentName() %></td>  <td><%= student.getYear() %></td>  </tr>  <% } %>  </tbody>  </table>  <my:pagination pageSize="<%= pageSize %>" recordCount="<%= recordCount %>" queryStringName="pg" />  </div>  </body>  </html> |

## 실행 과정 설명



(1)

studentList4.jsp를 실행하면,

웹브라우저에서 서버로, http://localhost:8080/lecture1/jdbc2/studentList4.jsp URL이 요청된다.

이 요청에 request parameter 데이터는 들어있지 않다.

(2)

서버에서 studentList4.jsp 파일이 실행되고,

StudentDAO2.findByName("", 1, 10) 메소드가 호출된다.

이 메소드는 1번 페이지에 출력할 10 개의 학생 레코드를 리턴한다.

StudentDAO2.count("") 메소드가 호출된다.

이 메소드는 studnet 테이블에서, name LIKE '%' 조건식으로 조회한다.

이 조건식은 언제나 true 이다.

따라서, StudentDAO2.count("") 메소드 호출은, student 테이블의 전체 레코드 수를 리턴한다.

(3) 리턴된 10 개의 학생 레코드가 출력된다.

(4)

전체 레코드 수로부터 총 페이지 수가 계산되고, 페이지 번호 목록이 출력된다.

pagination 확장 태그에서 의해서, 10 개의 페이지 번호가 출력된다.

이 페이지 번호 각각은 a 태그이다. 이 a 태그들은 다음과 같다.

|  |
| --- |
| <li class='active'>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=1'>1</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=2'>2</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=3'>3</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=4'>4</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=5'>5</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=6'>6</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=7'>7</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=8'>8</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=9'>9</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=10'>10</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?pg=11'>&gt;</a>  </li> |

pagination 확장 태그는, 서버에 요청된 URL에서 pg 값을 바꿔가며, 위와 같은 10 개의 a 태그를 출력한다.

(5) 위 화면에서 2번 페이지 번호를 클릭하면,

<a href='/lecture1/jdbc2/studentList4.jsp?pg=2'>2</a> 태그가 클릭되고,

http://localhost:8080/lecture1/jdbc2/studentList4.jsp?pg=2 URL이 서버에 요청된다.

(6)

서버에서 studentList4.jsp 파일이 실행되고,

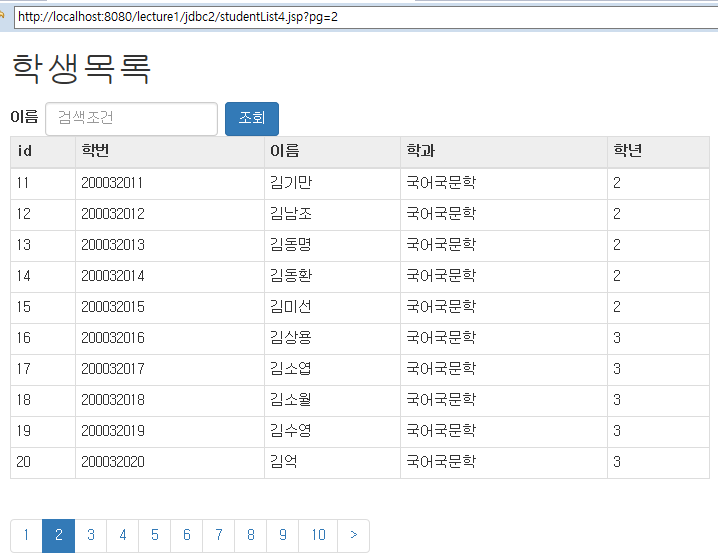
StudentDAO2.findByName("", 2, 10) 메소드가 호출된다.

이 메소드는 2번 페이지에 출력할 10 개의 학생 레코드를 리턴한다.

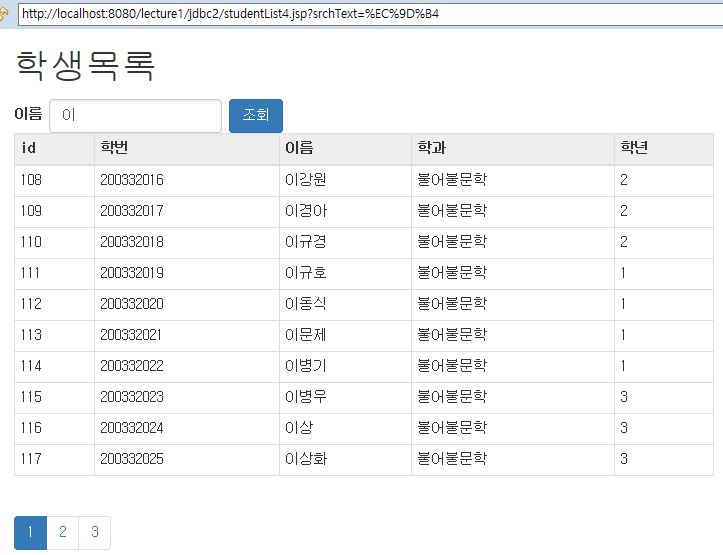
StudentDAO2.count("") 메소드 호출은, student 테이블의 전체 레코드 수를 리턴한다.

(7) 리턴된 10 개의 학생 레코드가 출력된다.

전체 레코드 수로부터 총 페이지 수가 계산되고, 페이지 번호 목록이 출력된다.



## 실행 과정 설명



(1)

조회조건 텍스트 박스에 "이"를 입력하고 조회버튼을 클릭하자.

이 버튼은 submit 버튼이다.

이 버튼을 감싸고 있는 form 태그에는 action 애트리뷰트가 없다.

action 애트리뷰트가 없는 form이 submit 될 때, 서버에 요청되는 URL은,

현재 웹브라우저창의 URL에서 query string을 제외한 부분이다.

|  |
| --- |
| http://localhost:8080/lecture1/jdbc2/studentList4.jsp |

GET 방식 요청이기 때문에, form의 input 태그에 입력된 내용이,

서버에 요청되는 URL의 query string에 추가된다.

|  |
| --- |
| http://localhost:8080/lecture1/jdbc2/studentList4.jsp?srchText=%EC%9D%B4 |

입력 폼에 입력된 한글은 자동으로 URL encoding 되어 query string에 추가된다.

(2)

서버에서 studentList4.jsp 파일이 실행되고,

StudentDAO2.findByName("이", 1, 10) 메소드가 호출된다.

이 메소드는 이름의 앞 부분이 "이"로 시작하는 학생 목록의 1번 페이지 출력할 10 개의 학생 레코드를 리턴한다.

StudentDAO2.count("이") 메소드가 호출된다.

이 메소드는 studnet 테이블에서, name LIKE '이%' 조건식으로 조회한다.

StudentDAO2.count("이") 메소드 호출은, student 테이블에서, 이름이 '이'로 시작하는 레코드 수를 리턴한다.

(3) 리턴된 10 개의 학생 레코드가 출력된다.

(4)

이름이 '이'로 시작하는 레코드 수로부터 총 페이지 수가 계산된다. 총 페이지 수는 3 이다.

pagination 확장 태그에서 의해서, 3 개의 페이지 번호가 출력된다.

이 페이지 번호 각각은 a 태그이다. 이 a 태그들은 다음과 같다.

|  |
| --- |
| <li class='active'>  <a href='/lecture1/jdbc2/studentList4.jsp?srchText=%EC%9D%B4&pg=1'>1</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?srchText=%EC%9D%B4&pg=2'>2</a>  </li>  <li class=''>  <a href='/lecture1/jdbc2/studentList4.jsp?srchText=%EC%9D%B4&pg=3'>3</a>  </li> |

pagination 확장 태그는, 서버에 요청된 URL에서 pg 값을 바꿔가며, 위와 같은 10 개의 a 태그를 출력한다.

방금 서버에 요청된 URL은 "http://localhost:8080/lecture1/jdbc2/studentList4.jsp?srchText=%EC%9D%B4" 이다.

(5) 위 화면에서 2번 페이지 번호를 클릭하면,

<a href='/lecture1/jdbc2/studentList4.jsp?srchText=%EC%9D%B4&pg=2'>2</a> 태그가 클릭되고,

http://localhost:8080/lecture1/jdbc2/studentList4.jsp?srchText=%EC%9D%B4&pg=2 URL이 서버에 요청된다.

(6)

서버에서 studentList4.jsp 파일이 실행되고,

StudentDAO2.findByName("이", 2, 10) 메소드가 호출된다.

이 메소드는 이름의 앞 부분이 "이"로 시작하는 학생 목록의 2번 페이지에 출력할 10 개의 학생 레코드를 리턴한다.

StudentDAO2.count("이") 메소드 호출은, student 테이블에서, 이름이 '이'로 시작하는 레코드 수를 리턴한다.

(7) 리턴된 10 개의 학생 레코드가 출력된다.

이름이 '이'로 시작하는 레코드 수로부터 총 페이지 수가 계산된다. 총 페이지 수는 3 이다.

pagination 확장 태그에서 의해서, 3 개의 페이지 번호가 출력된다.