Class 20230116

> 💡 **단어 공부**
>
> https://drive.google.com/file/d/13q1mK8XfaPJbKLBpaJILZbsRkx57N5FA/view?usp=drivesdk

1교시 - (9:00 ~ 9:50)

Connection Pool - Basic

- JDBC를 사용할 때 가장 많이 리소스 측 자원이 소모되는 부분이 DB 연동에 필요한 Connection 객체를 생성하는 부분임.
- 지금까지 방법들은 모두 JSP에서 SQL 구문을 수행하기 위해 Connection 객체를 생성하고 사용 후 제거하는 과정을 반복해왔음. → 접속자가 많아질 경우 시스템의 성능을 급격하게 저하시킴.
- 따라서 이러한 문제점을 해결하기 위해 Connection Pool을 이용.
- 사용자가 접속할때마다 매번 새로운 Connection 객체를 생성하는 것이 아니라 일정 개수의 Connection 객체를 미리 생성 해 놓고 사용자의 요청이 있을 때마다 가용한 객체를 할당하고 다시 회수하는 방식.

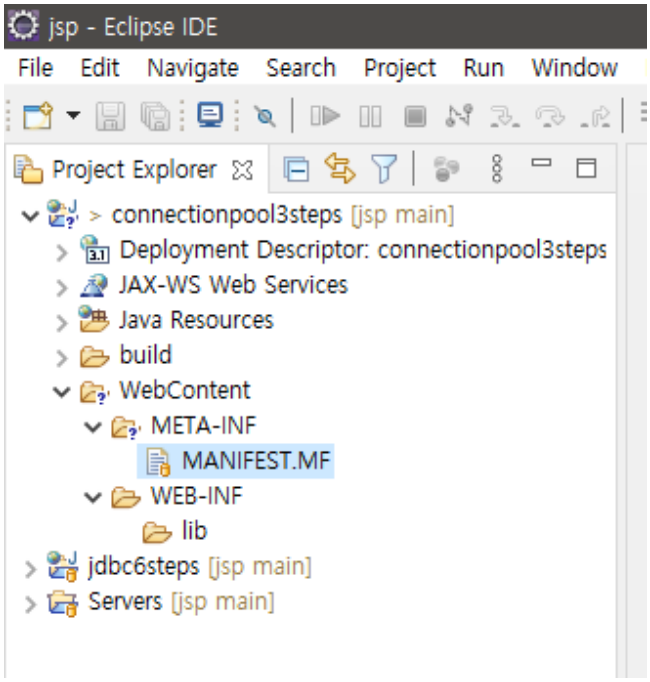
Connection Pool 설정

1. 커넥션 풀 설정 정의 - context.xml
2. 정의된 내용으로 실제 DB와 연결 해주는 객체를 생성하기 위한 클래스 작성 - ConnectionPool.java

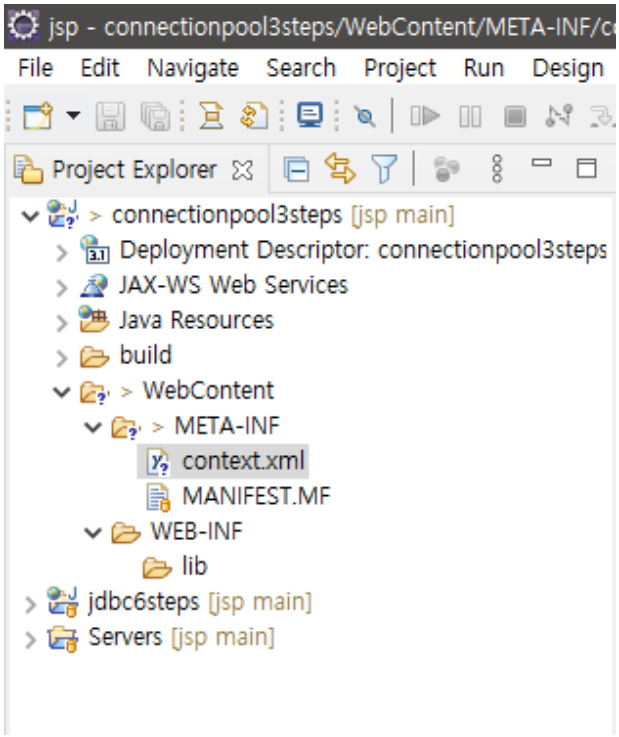
3. JDBC connector driver

1. context.xml

- 데이터 베이스에 대한 커넥션 풀을 사용하기 위한 설정 정의.
- 위치는 WebContent → META-INF → context.xml



▼ context.xml 생성




```
<?xml version="1.0" encoding="UTF-8"?>
<Context>
  <Resource name="jdbc/univ"      --> univ 사용할 디비명
  auth="Container"
  type="javax.sql.DataSource"
  driverClassName="com.mysql.jdbc.Driver"     --> 연결할 DB mysql, oracle ....
  url="jdbc:mysql://localhost:3306/univ?serverTimezone=UTC"
  username="root"  --> DB 아이디 ( 호스팅 업체에 업로드시에는 변경 )
  password="0000"  --> DB 패스워드 ( 호스팅 업체에 업로드시에는 변경 )
  maxTotal="16"  --> 미리 생성할 커넥션의 갯수
  maxIdle="4"    --> 최저 유지 커넥션 갯수
  maxWaitMillis="-1"/>    --> 항상 -1, 기다리는 시간!! 기다리지 않고 바로바로 처리.
</Context>

// 사용할 때 DB만 바꿔주면 됨.
// ?serverTimezone=UTC 특정 서버에서 타임존 설정을 하지 않으면 동작하지 않을때가 있다.
```

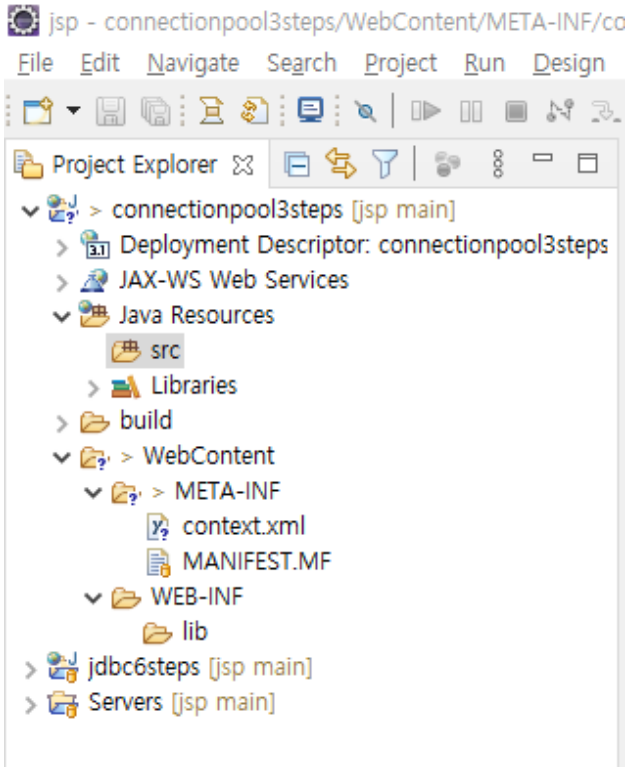
context.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Context>
    <Resource name="jdbc/univ"
    auth="Container"
    type="javax.sql.DataSource"
    driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost:3306/univ?serverTimezone=UTC"
    username="root"
    password="0000"
    maxTotal="16"
    maxIdle="4"
    maxWaitMillis="-1"/>
</Context>
```

2교시 - (10:00 ~ 10:50)

2. ConnectionPool.java

- 정의된 내용으로 실제 DB와 연결 해주는 객체를 생성하기 위한 클래스 작성
- 위치 : src → util 패키지 생성

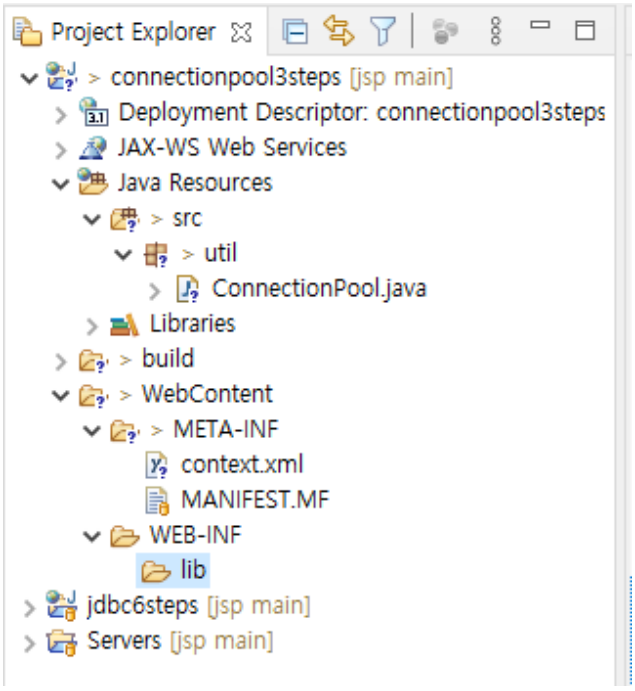


▼ ConnectionPool.java 생성

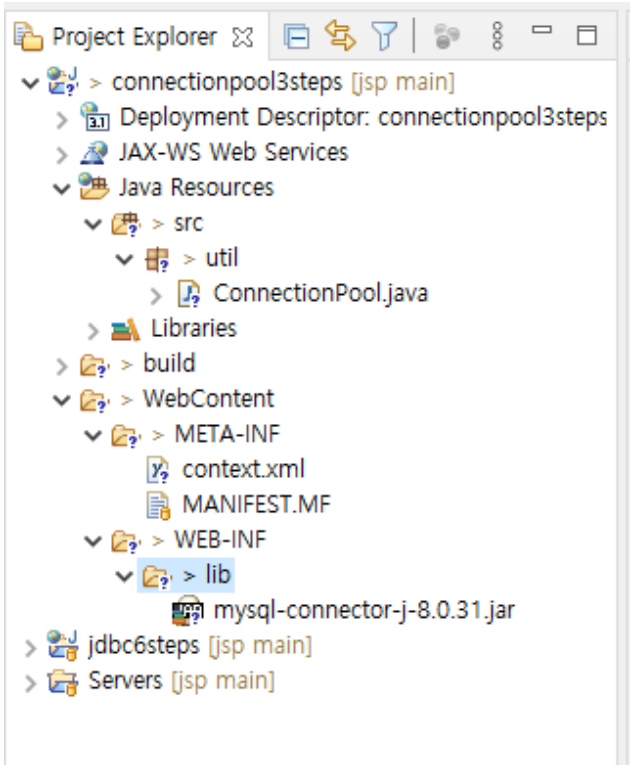
```
package util;

import java.sql.Connection;
import java.sql.SQLException;

import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;

public class ConnectionPool {
  private static DataSource _ds = null;

  public static Connection get() throws NamingException, SQLException {
    if(_ds == null) {
      _ds = (DataSource) (new InitialContext()).lookup("java:comp/env/jdbc/univ");
    }                                                          --> DB명만 바뀐다.
    return _ds.getConnection();
  }
}
```

3.JDBC connector driver

- 위치 : Webcontext → WEB-INF → lib



- 커넥터 연결 파일 옮기기



위 3단계로 Connection Pool 설정 완료

Connection Pool 적용

> 💡 항상 DB 설계부터 시작하자…

- 지난주 만든 DB와 테이블을 이용하여 Connection Pool에만 집중!

4. DTO (Data Transfer Object) → VO와 비슷한 개념

- DTO는 DB에서 데이터를 꺼낼때만 사용.
- DTO는 SETTER GETTER 존재 → SETTER 생략 가능(선택)
- VO는 GETTER만 존재
- DTO 파일은 데이터베이스의 테이블의 필드와 일대일 매칭이 되게 설계한다.
- jdbc 패키지 → StudentDTO 생성

> 데이터의 필드명으로 변수를 private 키워드로 생성하고 게터와 세터 그리고 생성자를 만듦.

```
package jdbc;

public class StudentDTO {

  private String hakbun;
  private String name;
  private String dept;
  private String addr;

  public String getHakbun() {
    return hakbun;
  }
  public void setHakbun(String hakbun) {
```

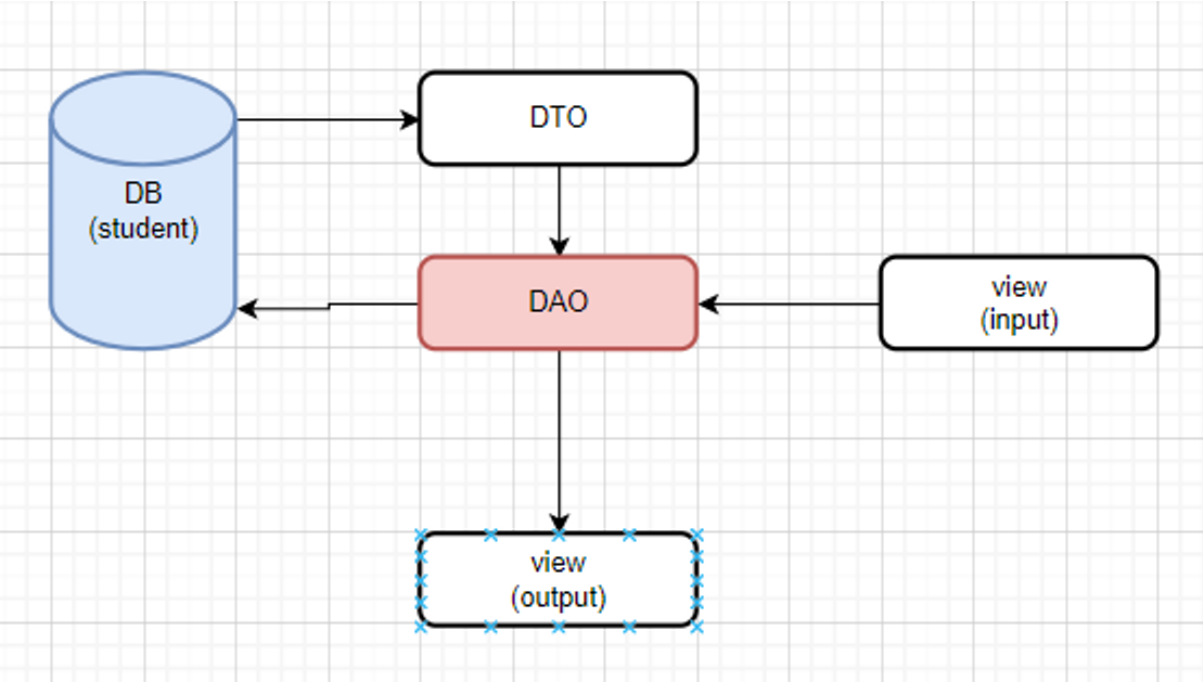
```
    this.hakbun = hakbun;
  }
  public String getName() {
    return name;
  }
  public void setName(String name) {
    this.name = name;
  }
  public String getDept() {
    return dept;
  }
  public void setDept(String dept) {
    this.dept = dept;
  }
  public String getAddr() {
    return addr;
  }
  public void setAddr(String addr) {
    this.addr = addr;
  }


  public StudentDTO(String hakbun, String name, String dept, String addr) {
    super();
    this.hakbun = hakbun;
    this.name = name;
    this.dept = dept;
    this.addr = addr;
  }
}
```

5. DAO (Data Access Object)

- 실제 DB와 연결되는 메서드 등과 SQL 쿼리등을 작성

```
package jdbc;

import java.sql.*;

import javax.naming.NamingException;

import util.ConnectionPool;


public class StudentDAO {

  //테이블에 데이터를 입력하는 메소드
  public static int insert(String hakbun, String name, String dept, String addr)
        throws NamingException, SQLException {

    // C R U D

    Connection conn = ConnectionPool.get(); //커넥션 풀 사용
```

```
    String sql = "INSERT INTO student VALUES(?, ?, ?, ?)";

    PreparedStatement pstmt = conn.prepareStatement(sql);
      pstmt.setString(1, hakbun);
      pstmt.setString(2, name);
      pstmt.setString(3, dept);
      pstmt.setString(4, addr);

      int result = pstmt.executeUpdate(); // SQL 구문 실행 성공시 1 실패시 0으로 나온다.

      return result;
  }
}
```



3교시 - (11:00 ~ 11:50)

6. View

- TBForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
  <form action="TBInsert.jsp" method="post">
    학번 <input type="text" name="hakbun"><br>
    이름 <input type="text" name="name"><br>
    부서 <input type="text" name="dept"><br>
    주소 <input type="text" name="addr"><br>
    <button>제출</button>
  </form>
</body>
</html>
```

- TBInsert.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%--        <!-- Step 1 import Packages  -->
<%@ page import="java.sql.*" %>  --%>
<%@page import="jdbc.*"%>

<% // 전송 받는 데이터 한글 처리
  request.setCharacterEncoding("UTF-8");
%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<%
    /* // Step 2 load JDBC Driver
  try{
  Class.forName("com.mysql.jdbc.Driver"); // JDBC 드라이버 가져오기
  } catch(ClassNotFoundException e){
    out.print("JDBC Driver loading error<br>" + e.getMessage());
  }

  //  Step 3 create Connection Object
```

```
  Connection conn = null;

  try{
    conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/univ",
                               "root", "0000");   // JDBC 드라이버 연결
  } catch(SQLException e){
    out.print("Connection Object error<br>" + e.getMessage());
  } */


  // Step 4 create Statement Object

  //테이블을 JSP로 생성
  //테이블명 student (univ db 이용)
  //hakbun, name, dept, addr

  /* String hakbun = "2222";
  String name = "Kim";
  String dept = "컴공";
  String addr = "인천"; */

  String hakbun = request.getParameter("hakbun");
  String name = request.getParameter("name");
  String dept = request.getParameter("dept");
  String addr = request.getParameter("addr");



  /* String sql = "INSERT INTO student VALUES(?, ?, ?, ?)";


  PreparedStatement pstmt = conn.prepareStatement(sql); //구문 생성
  pstmt.setString(1, request.getParameter("hakbun"));
  pstmt.setString(2, request.getParameter("name"));
  pstmt.setString(3, request.getParameter("dept"));
  pstmt.setString(4, request.getParameter("addr")); */

  // Step 5 excute SQL Query

  //pstmt.executeUpdate();

  // Step 6 close Connection

  //pstmt.close();
  //conn.close();

  int result = StudentDAO.insert(hakbun, name, dept, addr);

  if(result == 1){
    out.print("등록 성공");
  } else{
    out.print("등록 실패");
  }

%>

</body>
</html>
```

4교시 - (12:00 ~ 12:50)

회원 조회 DAO

```
//회원 조회
  public static ArrayList<StudentDTO> getList() throws NamingException, SQLException {

    String sql = "SELECT * FROM student";

    Connection conn = ConnectionPool.get();

    PreparedStatement pstmt = conn.prepareStatement(sql);

    ResultSet rs = pstmt.executeQuery();

    ArrayList<StudentDTO> students = new ArrayList<StudentDTO>();

    while(rs.next()) {
      students.add(new StudentDTO(rs.getString(1),
                  rs.getString(2),
                  rs.getString(3),
                  rs.getString(4)));
    }

    return students;

  }
```

- TBList.jsp

```
<%@page import="jdbc.StudentDTO"%>
<%@page import="java.util.ArrayList"%>
<%@page import="jdbc.StudentDAO"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>


<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="UTF-8">
<title>학생 목록</title>
</head>
<body>
<%
  ArrayList<StudentDTO> students = StudentDAO.getList();

  for(StudentDTO student : students){

%>
  <!-- 화면에 뿌려 주기  -->
  <%= student.getHakbun() %>|
  <%= student.getName() %>|
  <%= student.getDept() %>|
  <%= student.getAddr() %><br>

<%
  } // END foreach()

%>

</body>
</html>
```

- 회원 상세보기 DAO

```
//회원 상세보기
    public static StudentDTO getDetail(String hakbun) throws NamingException, SQLExcepti
on {

      String sql = "SELECT * FROM student WHERE hakbun =?";

      Connection conn = ConnectionPool.get();

      PreparedStatement pstmt = conn.prepareStatement(sql);
      pstmt.setString(1, hakbun);

      ResultSet rs = pstmt.executeQuery();

      StudentDTO student = null;

      if(rs.next()) {
        String name = rs.getString(2);
        String dept = rs.getString(3);
        String addr = rs.getString(4);
        student = new StudentDTO(hakbun, name, dept, addr);

      }

      /*
       * rs.next();
       *
       * String name = rs.getString(2); String dept = rs.getString(3); String addr =
       * rs.getString(4);
```

```
        *
        * StudentDTO student = new StudentDTO(hakbun, name, dept, addr);
        */


        return student;

    }
```

- 학번을 누르면 상세보기가 되는 TBList2.jsp

```
<%@page import="jdbc.StudentDTO"%>
<%@page import="java.util.ArrayList"%>
<%@page import="jdbc.StudentDAO"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>


<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>학생 목록</title>
</head>
<body>
<%
  ArrayList<StudentDTO> students = StudentDAO.getList();

  for(StudentDTO student : students){

%>
  <!-- 화면에 뿌려 주기  -->
  <a href="TBDetail.jsp?hakbun=<%= student.getHakbun() %>"><%= student.getHakbun() %></a
> |
  <%= student.getName() %><br>

<%
  } // END foreach()

%>

</body>
</html>
```

- TBDetail.jsp 생성

```
<%@page import="jdbc.StudentDTO"%>
<%@page import="java.util.ArrayList"%>
<%@page import="jdbc.StudentDAO"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
```

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>학생 상세보기</title>
</head>
<body>
<%
  String hakbun = request.getParameter("hakbun");
  StudentDTO student = StudentDAO.getDetail(hakbun);


%>
  <!-- 화면에 뿌려 주기  -->
  <%= student.getHakbun() %> |
  <%= student.getName() %> |
  <%= student.getDept() %> |
  <%= student.getAddr() %><br>

<%

%>

</body>
</html>
```

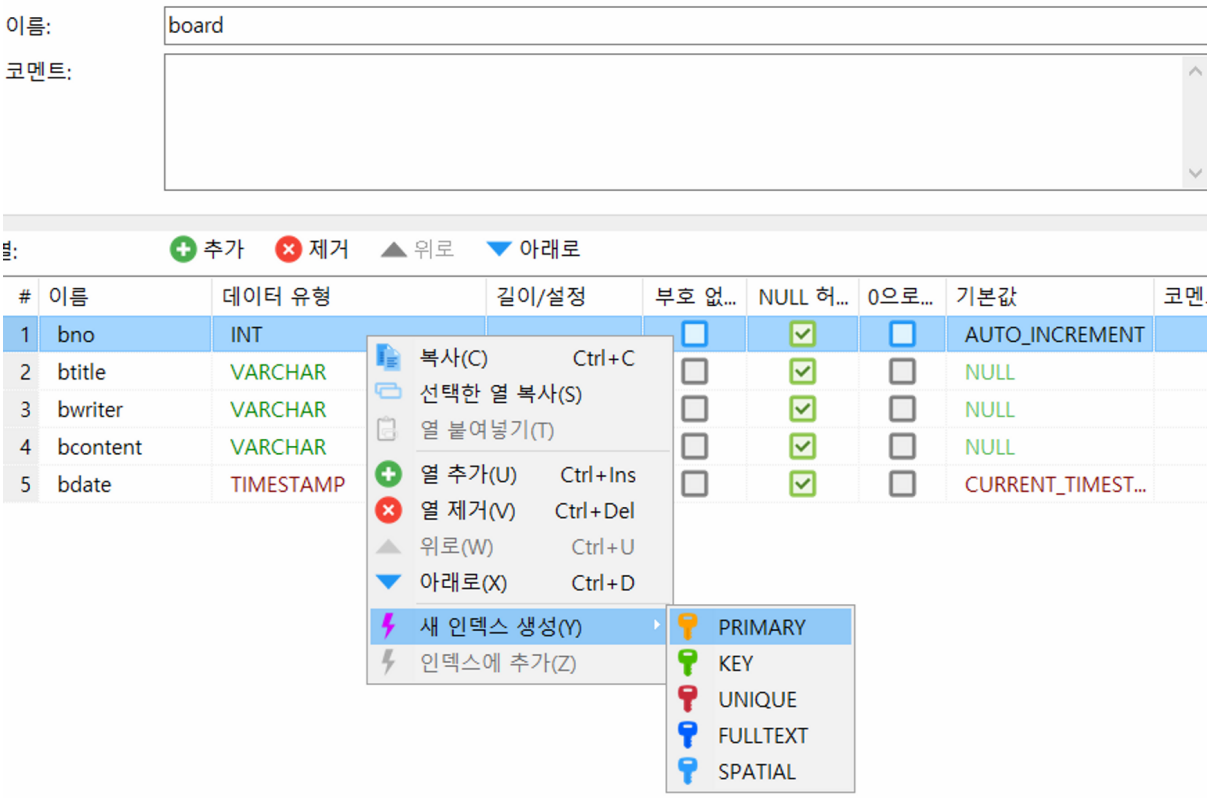
5교시 - (140:00 ~ 14:50)

DB 설계

테이블명 : board

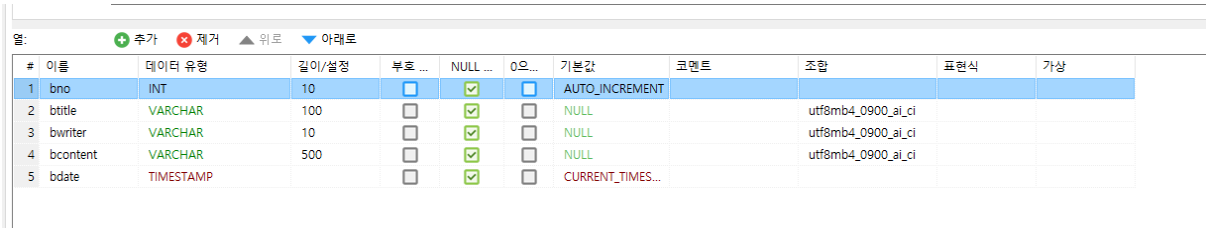
- 글번호 bno   100
- 제목   btitle   100
- 작성자 bwriter   10
- 내용   bcontent   500
- 날짜   bdate   x

기본키 설정



bno 자동증가 번호 넣기

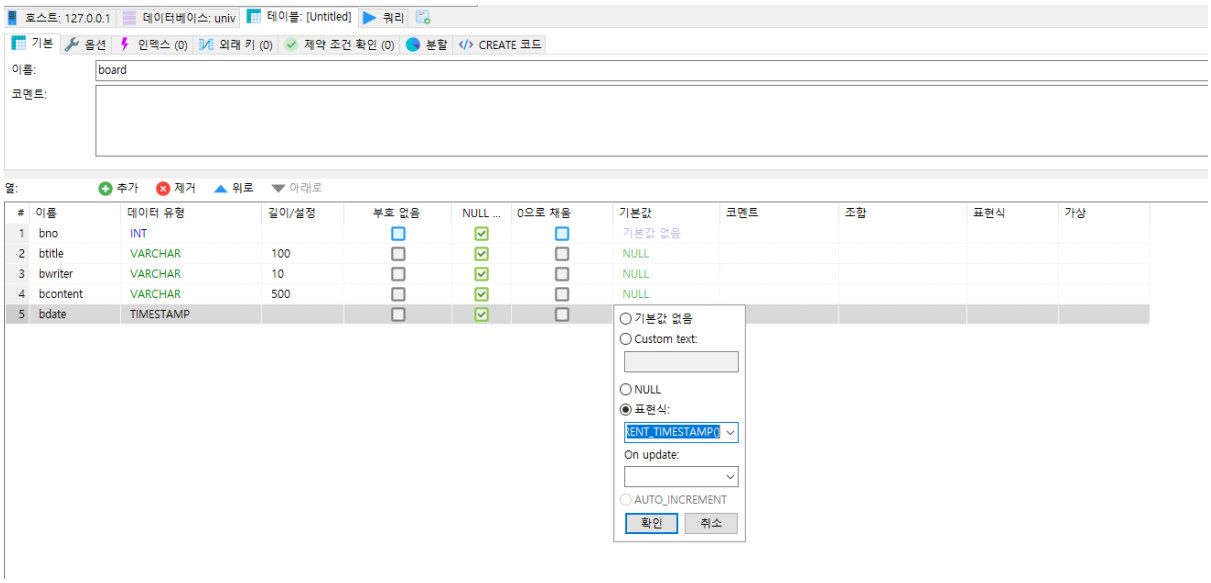
- 데이터 유형 : INT
- 기본값 → AUTO_INCREMENT



테이블에 데이터가 입력될때 자동으로 시간 입력

- 데이터 유형 : TIMESTAMP

- 기본값 → 표현식 → CURRENT_TIMESTAMP( )



BoardExam 프로젝트 생성 및 BoardDAO 작성

게시물 등록 DAO

```
//게시물 등록
  public static int insert(String btitle, String bwriter, String bcontent) throws Naming
Exception, SQLException {

    String sql = "INSERT INTO board(btitle, bwriter, bcontent) VALUES(?, ?, ?)";

    Connection conn = ConnectionPool.get();

    PreparedStatement pstmt = conn.prepareStatement(sql);
    pstmt.setString(1, btitle);
    pstmt.setString(2, bwriter);
    pstmt.setString(3, bcontent);

    int result = pstmt.executeUpdate();

    return result;
```

6교시 - (15:00 ~ 15:50)

BoardInsert.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="jdbc.*" %>
<%
  request.setCharacterEncoding("UTF-8");  // 전송 받는 데이터 한글 처리
%>

<%

  String btitle = request.getParameter("btitle");
  String bwriter = "작성자"; //작성자는 로그인한 세션값을 읽어서 사용한다.
  String bcontent = request.getParameter("bcontent");

  int result = BoardDAO.insert(btitle, bwriter, bcontent);

  if(result == 1){
    out.print("등록 성공");
  } else{
    out.print("등록 실패");
  }

%>
```

BoardForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.c
ss" rel="stylesheet" integrity="sha384-GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI
1aFkw7cmDA6j6gD" crossorigin="anonymous">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>

<div class="container">
<form action="BoardInsert.jsp">
  <div class="mb-3">
    <label for="exampleFormControlInput1" class="form-label">제목</label>
    <input type="text" class="form-control" name="btitle">
  </div>
  <div class="mb-3">
```

```
    <label for="exampleFormControlTextarea1" class="form-label">내용</label>
    <textarea class="form-control" name="bcontent" rows="3"></textarea>
  </div>
  <button type="submit" class="btn btn-primary">등록</button>
</form>
</div>

</body>
</html>
```

게시물 조회 DAO

```
//게시물 조회
  public static ArrayList<BoardDTO> getList() throws NamingException, SQLException {

    String sql = "SELECT * FROM board";

    Connection conn = ConnectionPool.get();

    PreparedStatement pstmt = conn.prepareStatement(sql);

    ResultSet rs = pstmt.executeQuery();

    ArrayList<BoardDTO> boards = new ArrayList<BoardDTO>();

    while(rs.next()) {
      boards.add(new BoardDTO(rs.getString(1),
                    rs.getString(2),
                    rs.getString(3),
                    rs.getString(4),
                    rs.getString(5)));
    }

    return boards;

  }
```

BoardList.jsp

```
<%@page import="jdbc.BoardDAO"%>
<%@page import="jdbc.BoardDTO"%>
<%@page import="java.util.ArrayList"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.c
```

```
ss" rel="stylesheet" integrity="sha384-GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI
1aFkw7cmDA6j6gD" crossorigin="anonymous">
<head>
<meta charset="UTF-8">
<title>게시물 목록</title>
</head>
<body>
<div class="container">
<table class="table table-hover">
  <thead>
    <tr>
      <th scope="col">번호</th>
      <th scope="col">제목</th>
      <th scope="col">작성자</th>
      <th scope="col">날짜</th>
    </tr>
  </thead>
  <tbody>
<%
  ArrayList<BoardDTO> boards = BoardDAO.getList();

  for(BoardDTO board : boards){

%>
  <!-- 화면에 뿌려 주기  -->

    <tr>
      <th scope="row"><%=board.getBno() %></th>
      <td><%=board.getBtitle() %></td>
      <td><%=board.getBwriter() %></td>
      <td><%=board.getBdate() %></td>
    </tr>

<%
  } //END foreach()
%>
  </tbody>
</table>
</div>
</body>
</html>
```

7교시 - (16:00 ~ 16:50)

SummerNote 적용

- 주의사항

1. DB에 필드 사이즈를 크게 LONGTEXT로 설정

2. form method를 post로 변경

BoardFormSummer.jsp 생성

- 부트스트랩 4버전 사용

BoardFormSummer5.jsp 생성

- 부트스트랩 5버전이 안됨.

BoardFormSummerWO.jsp 생성

- 부트스트랩 안쓰는 버전을 COPY 하고 위에 부트스트랩 5 링크만 걸어준다.

8교시 - 정리(17:00 ~ 17:50)

모바일 화면 보기 설정

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```