

8장 마이바티스 프레임워크

강사 김영석

A top-down view of a wooden desk. On the desk, there is a silver laptop with a black keyboard, a pair of black-rimmed glasses, a white coffee cup with a yellow handle, and a small green succulent in a pot. The word "CONTENT" is written in large, white, sans-serif capital letters over the desk surface.

CONTENT

1

마이바티스란?

2

마이바티스 이용해 회원 기능
실습하기

3

마이바티스 이용해 회원 정보
CRUD 실습하기

4

마이바티스의 동적 SQL 문 사용
하기

1. 마이바티스란?

✓ 기존 JDBC의 문제점

- connection 이 Statement 객체를 생성하고 SQL문을 전송하면 결과를 반환하고 close 과정을 거치게 된다.
- SQL문이 프로그래밍 코드에 섞여 코드를 더욱 복잡하게 만들어 사용 및 유지 보수의 어려움이 생긴다.

✓ 마이바티스 프레임워크의 특징

- SQL 실행 결과를 자바 빈즈 또는 Map 객체에 매핑해 주는 Persistence 솔루션으로 관리한다. 즉 SQL문을 소스 코드가 아닌 XML 파일로 만들어 소스 코드와 분리 처리한다.
- 데이터소스(DataSource) 기능과 트랜잭션 처리 기능을 제공한다.



✓ 마이바티스의 동작 과정

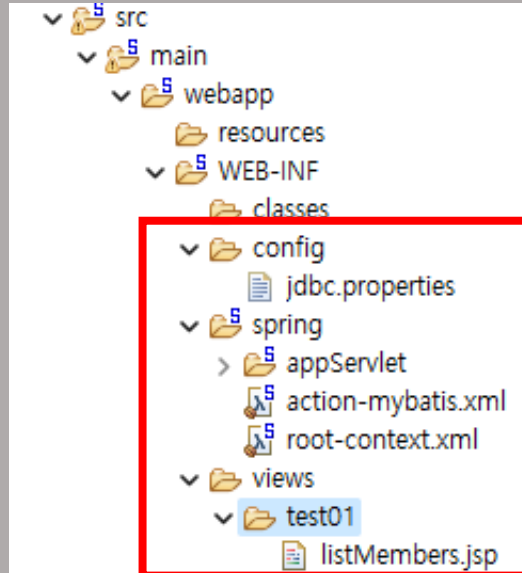
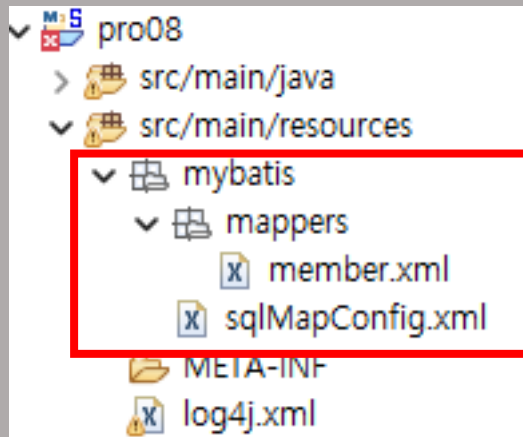
- ① SqlMapConfig.xml에 각 기능별로 실행할 SQL문을 SqlMap.xml에 미리 작성한 후 등록한다.
- ② 애플리케이션에서 데이터베이스와 연동하는 데 필요한 데이터를 각각의 매개변수에 저장한 후 마이바티스에 전달한다.
- ③ 애플리케이션에서 요청한 SQL문을 SqlMap.xml에서 선택한다.
- ④ 전달한 매개변수와 선택한 SQL문을 결합한다.
- ⑤ 매개변수와 결합된 SQL문을 DBMS에서 실행한다.
- ⑥ DBMS에서 반환된 데이터를 애플리케이션에서 제공하는 적당한 매개변수에 저장한 후 반환한다.

2. 마이바티스 이용해 회원 기능 실습하기

```
pro07/pom.xml
106 <dependency>
107   <groupId>javax.servlet</groupId>
108   <artifactId>jstl</artifactId>
109   <version>1.2</version>
110 </dependency>
111
112 <!-- 마이바티스 -->
113 <dependency>
114   <groupId>org.mybatis</groupId>
115   <artifactId>mybatis</artifactId>
116   <version>3.4.6</version>
117 </dependency>
118 <dependency>
119   <groupId>org.mybatis</groupId>
120   <artifactId>mybatis-spring</artifactId>
121   <version>1.3.2</version>
122 </dependency>
123 <dependency>
124   <groupId>org.springframework</groupId>
125   <artifactId>spring-jdbc</artifactId>
126   <version>${org.springframework-version}</version>
127 </dependency>
128
129 <!-- 스프링 트랜잭션 -->
130 <dependency>
131   <groupId>org.springframework</groupId>
132   <artifactId>spring-tx</artifactId>
133   <version>${org.springframework-version}</version>
134 </dependency>
135
136 <!-- 오라클 드라이버 -->
137 <dependency>
138   <groupId>com.oracle.jdbc</groupId>
139   <artifactId>ojdbc8</artifactId>
140   <version>19.3.0.0</version>
141 </dependency>
```

✓ 마이바티스 설정 파일

설정 파일	기능
SqlMapConfig.xml	데이터베이스 연동 시 반환되는 값을 저장할 빈이나 트랜잭션, 데이터 소스 등 마이바티스 관련 정보를 설정한다.
member.xml	회원 정보 관련 SQL문을 설정한다.



파일을 생성할 때 Other → XML → XML File 을 선택한다.
mabatis 패키지를 만들고
sqlMapConfig.xml 을 만들고 mappers 패키지를 만들면 편하다.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springfra
5 <bean id="propertyPlaceholderConfigurer"
6     class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
7     <property name="locations">
8         <value>/WEB-INF/config/jdbc.properties</value>
9     </property>
10 </bean>
11 <bean id="dataSource" class="org.apache.ibatis.datasource.pooled.PooledDataSource">
12     <property name="driver" value="${jdbc.driverClassName}"></property>
13     <property name="url" value="${jdbc.url}"></property>
14     <property name="username" value="${jdbc.username}"></property>
15     <property name="password" value="${jdbc.password}"></property>
16 </bean>
17 <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
18     <property name="dataSource" ref="dataSource" />
19     <property name="configLocation"
20         value="classpath:mybatis/sqlMapConfig.xml" />
21     <property name="mapperLocations" value="classpath:mybatis/mappers/*.xml" />
22 </bean>
23
24 <bean id="sqlSession" class="org.mybatis.spring.SqlSessionTemplate">
25     <constructor-arg index="0" ref="sqlSessionFactory"></constructor-arg>
26 </bean>
27 </beans>
```

jdbc.properties


```
1 jdbc.driverClassName=oracle.jdbc.driver.OracleDriver
2 jdbc.url=jdbc:oracle:thin:@localhost:1521:XE
3 jdbc.username=system
4 jdbc.password=oracle
```

SqlMapConfig.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE configuration PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
3     "http://mybatis.org/dtd/mybatis-3-config.dtd">
4
5 <configuration>
6     <typeAliases>
7         <typeAlias type="com.test.pro07.ex01.MemberVO" alias="memberVO" />
8     </typeAliases>
9 </configuration>
```

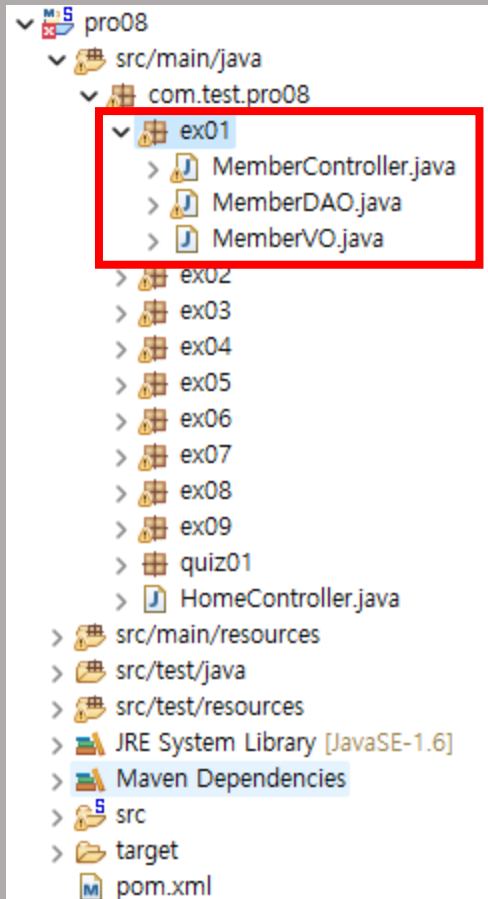



```
member.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
3     "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
4
5 <mapper namespace="mapper.member">
6     <resultMap type="memberVO" id="memResult">
7         <result property="id" column="id"/>
8         <result property="pwd" column="pwd"/>
9         <result property="name" column="name"/>
10        <result property="email" column="email"/>
11        <result property="joinDate" column="joinDate"/>
12    </resultMap>
13
14    <select id="selectAllMemberList" resultMap="memResult">
15        <![CDATA[
16            select * from t_member order by joinDate desc
17        ]]>
18    </select>
19 </mapper>
```




```
web.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/xsds/web-app_2_5.xsd">
5
6     <!-- The definition of the Root Spring Container shared by all modules -->
7     <context-param>
8         <param-name>contextConfigLocation</param-name>
9         <param-value>
10             /WEB-INF/spring/root-context.xml
11             /WEB-INF/spring/action-mybatis.xml
12         </param-value>
13     </context-param>
14
```

3. 마이바티스 이용해 회원 정보 CRUD 실습하기



```
MemberController.java
1 package com.test.pro08.ex01;
2
3 import java.util.List;
12
13 //@Controller
14 @RequestMapping("/test01")
15 public class MemberController {
16     @Autowired
17     MemberDAO dao;
18
19     @RequestMapping(value="/mem1.do")
20     public ModelAndView listMembers(HttpServletRequest request,
21                                     HttpServletResponse response) throws Exception{
22         List<MemberVO> membersList = dao.selectAllList();
23         ModelAndView mav = new ModelAndView("test01/listMembers");
24         mav.addObject("membersList", membersList);
25         return mav;
26     }
27 }
```



```
MemberDAO.java ✕
1 package com.test.pro08.ex01;
2
3 import java.util.List;
4
5
6 // @Repository
7
8 public class MemberDAO {
9     @Autowired
10     private SqlSession sqlSession;
11
12     public List<MemberVO> selectAllList() {
13         // TODO Auto-generated method stub
14         List<MemberVO> membersList =
15             sqlSession.selectList("mapper.member.selectAllMemberList");
16         return membersList;
17     }
18 }
19
20 }
```



```

1 package com.test.pro08.ex01;
2
3 import java.sql.Date;
4
5 public class MemberVO {
6     private String id;
7     private String pwd;
8     private String name;
9     private String email;
10    private Date joinDate;
11    public String getId() {
12        return id;
13    }
14    public void setId(String id) {
15        this.id = id;
16    }
17    public String getPwd() {
18        return pwd;
19    }
20    public void setPwd(String pwd) {
21        this.pwd = pwd;
22    }
23    public String getName() {
24        return name;
25    }
26    public void setName(String name) {
27        this.name = name;
28    }

```

```

5 <%
6     request.setCharacterEncoding("UTF-8");
7 %>
8 <html>
9 <head>
10 <meta charset="UTF-8">
11 <title>회원 정보 출력창</title>
12 </head>
13 <body>
14 <table border="1" align="center" width="80%">
15     <tr align="center" bgcolor="lightgreen">
16         <td><b>아이디</b></td>
17         <td><b>비밀번호</b></td>
18         <td><b>이름</b></td>
19         <td><b>이메일</b></td>
20         <td><b>가입일</b></td>
21     </tr>
22
23 <c:forEach var="member" items="${membersList}" >
24     <tr align="center">
25         <td>${member.id}</td>
26         <td>${member.pwd}</td>
27         <td>${member.name}</td>
28         <td>${member.email}</td>
29         <td>${member.joinDate}</td>
30     </tr>
31 </c:forEach>
32 </table>
33 </html>

```

회원 정보 출력창

localhost8080/pro08/test01/mem1.do

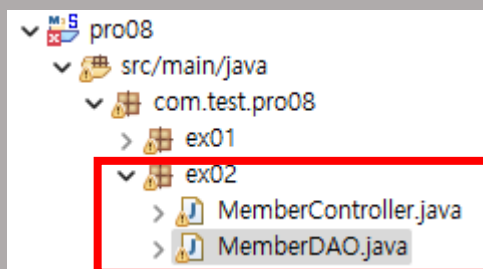
☆ □ 시크릿 모드

아이디	비밀번호	이름	이메일	가입일
ddd	111	ddd	ddd@test.com	2023-09-10
ccc	111	ccc	ccc@test.com	2023-09-10
bbb	111	111	bbb@test.com	2023-09-10
aaa	1111	aaa	aaa@test.com	2023-09-10
kim	1234	김유신	kim@test.com	2023-09-10
lee	1234	이순신	lee@test.com	2023-09-10
hong	1234	홍길동	hong@test.com	2023-09-10

✓ SqlSession 클래스에서 제공하는 여러가지 메서드

메서드	기능
List selectList(query_id)	id에 대한 select문을 실행한 후 여러 레코드를 List로 반환한다.
List selectList(query_id, 조건)	id에 대한 select문을 실행하면서 사용되는 조건도 전달한다.
T selectOne(query_id)	id에 대한 select문을 실행한 후 지정한 타입으로 한 개의 레코드를 반환한다.
T selectOne(query_id, 조건)	id에 대한 select문을 실행하면서 사용되는 조건도 전달한다.
Map<K,V> selectMap(query_id, 조건)	id에 대한 select문을 실행하면서 사용되는 조건도 전달한다. Map 타입으로 레코드를 반환한다.
int insert(query_id, Object obj)	id에 대한 insert문을 실행하면서 obj 객체의 값을 테이블에 추가한다.
int update(query_id, Object obj)	obj 객체의 값을 조건문의 수정 값으로 사용해 id에 대한 update 문을 실행한다.
int delete(query_id, Object obj)	obj 객체의 값을 조건문의 조건값으로 사용해 id에 대한 delete문을 실행한다.


✓ 회원의 ID와 비밀번호 조회



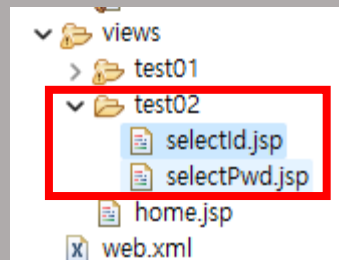
```
member.xml
25<select id="selectAllMemberList" resultMap="memResult">
26    <![CDATA[
27        select * from t_member order by joinDate desc
28    ]]>
29</select>
30<select id="selectId" resultType="String">
31    <![CDATA[
32        select id from t_member where id='hong'
33    ]]>
34</select>
35<select id="selectPwd" resultType="int">
36    <![CDATA[
37        select pwd from t_member where id='hong'
38    ]]>
39</select>
40</mapper>
```



```
MemberController.java
1 package com.test.pro08.ex02;
2
3 import javax.servlet.http.HttpServletRequest;
4
10
11 @Controller
12 @RequestMapping("/test02")
13 public class MemberController {
14     @Autowired
15     MemberDAO dao;
16
17     @RequestMapping(value="/mem1.do")
18     public ModelAndView selectId(HttpServletRequest request,
19                                 HttpServletResponse response) throws Exception{
20         String name = dao.selectId();
21         ModelAndView mav = new ModelAndView("test02/selectId");
22         mav.addObject("name", name);
23         return mav;
24     }
25
26     @RequestMapping(value="/mem2.do")
27     public ModelAndView selectPwd(HttpServletRequest request,
28                                   HttpServletResponse response) throws Exception{
29         int pwd = dao.selectPwd();
30         ModelAndView mav = new ModelAndView("test02/selectPwd");
31         mav.addObject("pwd", pwd);
32         return mav;
33     }
34 }
```

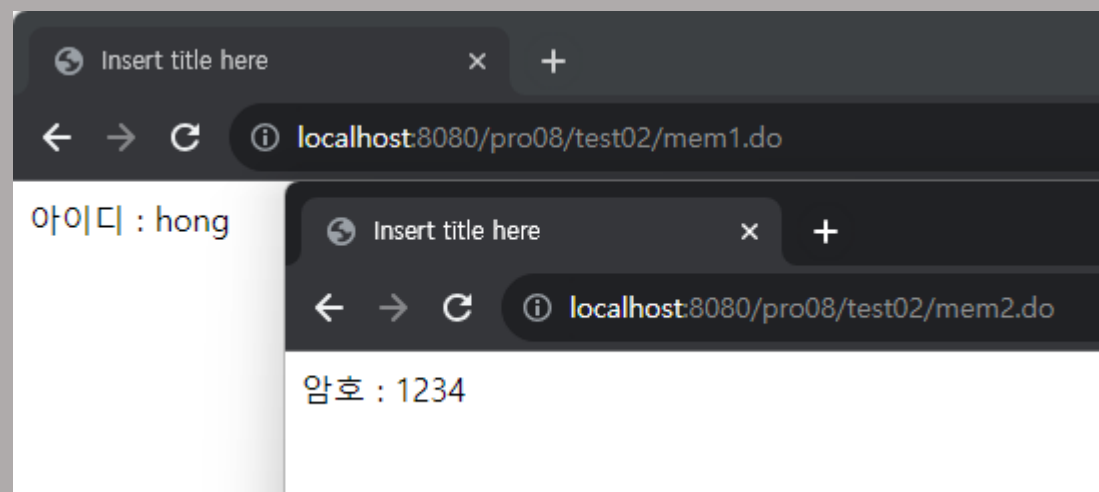


```
MemberDAO.java  ✖
1  package com.test.pro08.ex02;
2
3  import org.apache.ibatis.session.SqlSession;
4
5
6  @Repository
7
8  public class MemberDAO {
9      @Autowired
10     private SqlSession sqlSession;
11
12     public String selectId(){
13         String name = sqlSession.selectOne("mapper.member.selectId");
14         return name;
15     }
16
17     public int selectPwd(){
18         int pwd = sqlSession.selectOne("mapper.member.selectPwd");
19         return pwd;
20     }
21 }
22 }
```



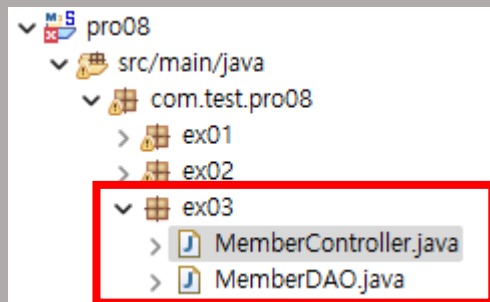
```
selectPwd.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8" %>
2     pageEncoding="UTF-8"
3     isELIgnored="false" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%
6     request.setCharacterEncoding("UTF-8");
7 %>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta charset="UTF-8">
12 <title>Insert title here</title>
13 </head>
14 <body>
15     암호 : ${pwd }
16 </body>
17 </html>
```

```
selectId.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8" %>
2     pageEncoding="UTF-8"
3     isELIgnored="false" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%
6     request.setCharacterEncoding("UTF-8");
7 %>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta charset="UTF-8">
12 <title>Insert title here</title>
13 </head>
14 <body>
15     아이디 : ${name }
16 </body>
17 </html>
```




✓ HashMap을 이용한 모든 회원 정보 조회

```
member.xml
10      <result property="email" column="email"></result>
11      <result property="joinDate" column="joinDate"></result>
12  </resultMap>
13  <resultMap type="java.util.HashMap" id="memMap">
14      <result property="id" column="id"></result>
15      <result property="pwd" column="pwd"></result>
16      <result property="name" column="name"></result>
17      <result property="email" column="email"></result>
18      <result property="joinDate" column="joinDate"></result>
19  </resultMap>
20  <select id="selectAllMemberMap" resultMap="memMap">
21      <![CDATA[
22          select * from t_member order by joinDate desc
23      ]]>
24  </select>
25  <select id="selectAllMemberList" resultMap="memResult">
26      <![CDATA[
```



```
MemberController.java 2
1 package com.test.pro08.ex03;
2
3 import java.util.List;
4
14
15 @Controller
16 @RequestMapping("/test03")
17 public class MemberController {
18     @Autowired
19     MemberDAO dao;
20
21     @RequestMapping(value="/mem1.do")
22     public ModelAndView selectAllMemberMap(HttpServletRequest request,
23         HttpServletResponse response) throws Exception{
24         List<MemberVO> membersList = dao.selectAllMemberMap();
25         ModelAndView mav = new ModelAndView("test03/listMembers");
26         mav.addObject("membersList", membersList);
27         return mav;
28     }
29 }
```



```
MemberDAO.java 2
1 package com.test.pro08.ex03;
2
3 import java.util.List;
4
10
11 @Repository
12 public class MemberDAO {
13     @Autowired
14     private SqlSession sqlSession;
15
16     public List<MemberVO> selectAllMemberMap() {
17         // TODO Auto-generated method stub
18         List<MemberVO> membersList =
19             sqlSession.selectList("mapper.member.selectAllMemberMap");
20         return membersList;
21     }
22 }
```

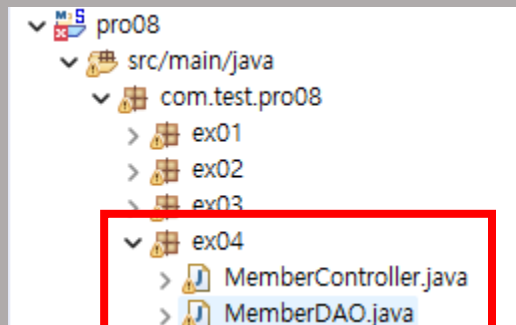
회원 정보 출력창

localhost:8080/pro08/test03/mem1.do

시크릿 모드

아이디	비밀번호	이름	이메일	가입일
ddd	111	ddd	ddd@test.com	2023-09-10 23:19:24.0
ccc	111	ccc	ccc@test.com	2023-09-10 23:16:40.0
bbb	111	111	bbb@test.com	2023-09-10 13:12:44.0
aaa	1111	aaa	aaa@test.com	2023-09-10 13:11:15.0
kim	1234	김유신	kim@test.com	2023-09-10 12:40:28.0
lee	1234	이순신	lee@test.com	2023-09-10 12:40:27.0
hong	1234	홍길동	hong@test.com	2023-09-10 12:40:26.0

✓ 조건 값으로 회원 정보 조회




```
member.xml
7      <result property="id" column="id"></result>
8      <result property="pwd" column="pwd"></result>
9      <result property="name" column="name"></result>
10     <result property="email" column="email"></result>
11     <result property="joinDate" column="joinDate"></result>
12 </resultMap>
13 <!-- <resultMap type="java.util.HashMap" id="memMap">
14     <result property="id" column="id"></result>
15     <result property="pwd" column="pwd"></result>
16     <result property="name" column="name"></result>
17     <result property="email" column="email"></result>
18     <result property="joinDate" column="joinDate"></result>
19 </resultMap>
20 <select id="selectAllMemberMap" resultMap="memMap">
21     <![CDATA[
22         select * from t_member order by joinDate desc
23     ]]>
24 </select> -->
25
26 <select id="selectAllMemberList" resultMap="memResult">
27     <![CDATA[
28         select * from t_member order by joinDate desc
29     ]]>
30 </select>
```


member.xml

```
37         select pwd from t_member where id='hong'
38     ]]>
39 </select>
40 <select id="selectMemberById" resultType="memberVO" parameterType="String">
41     <![CDATA[
42         select * from t_member where id=#{id}
43     ]]>
44 </select>
45 <select id="selectMemberByPwd" resultMap="memResult" parameterType="int">
46     <![CDATA[
47         select * from t_member where pwd=#{pwd}
48     ]]>
49 </select>
50 </mapper>
```

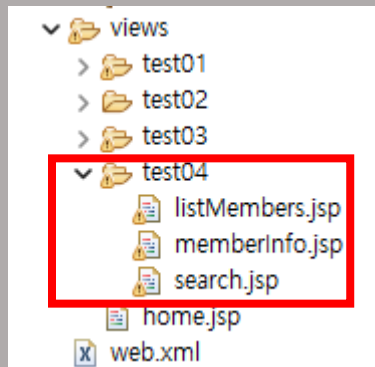
```
MemberController.java
1 package com.test.pro08.ex04;
2
3 import java.util.List;
14
15 @Controller
16 @RequestMapping("/test04")
17 public class MemberController {
18     @Autowired
19     MemberDAO dao;
20
21     @RequestMapping(value="/mem1.do")
22     public ModelAndView selectId(HttpServletRequest request,
23                                 HttpServletResponse response) throws Exception{
24         String action = request.getParameter("action");
25         String viewName = "";
26         ModelAndView mav = new ModelAndView();
27         if(action == null || action.equals("listMembers")) {
28             List<MemberVO> membersList = dao.selectAllMemberList();
29             mav.addObject("membersList", membersList);
30             viewName = "test04/listMembers";
31         } else if (action.equals("selectMemberById")) {
32             String id = request.getParameter("value");
33             MemberVO memberVO = dao.selectMemberById(id);
34             mav.addObject("member", memberVO);
35             viewName = "test04/memberInfo";
```



```
36         } else if (action.equals("selectMemberByPwd")) {
37             int pwd = Integer.parseInt(request.getParameter("value"));
38             List<MemberVO> membersList = dao.selectMemberByPwd(pwd);
39             mav.addObject("membersList", membersList);
40             viewName = "test04/listMembers";
41         }
42         mav.setViewName(viewName);
43         return mav;
44     }
45
46     @RequestMapping(value="/search.do")
47     public ModelAndView search(HttpServletRequest request,
48                             HttpServletResponse response) throws Exception{
49         ModelAndView mav = new ModelAndView("test04/search");
50         return mav;
51     }
52 }
```

```
MemberDAO.java
1 package com.test.pro08.ex04;
2
3 import java.util.List;
10
11 @Repository
12 public class MemberDAO {
13     @Autowired
14     private SqlSession sqlSession;
15
16     public List<MemberVO> selectAllMemberList() {
17         // TODO Auto-generated method stub
18         List<MemberVO> membersList =
19             sqlSession.selectList("mapper.member.selectAllMemberList");
20         return membersList;
21     }
22     public MemberVO selectMemberById(String id) {
23         MemberVO vo = sqlSession.selectOne("mapper.member.selectMemberById", id);
24         return vo;
25     }
26     public List<MemberVO> selectMemberByPwd(int pwd) {
27         List<MemberVO> membersList =
28             sqlSession.selectList("mapper.member.selectMemberByPwd", pwd);
29         return membersList;
30     }
31 }
```



```
search.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>회원 검색창</title>
8 </head>
9 <body>
10 <form action="/pro08/test04/mem1.do">
11   입력 : <input type="text" name="value">
12   <select name="action">
13     <option value="listMembers">전체</option>
14     <option value="selectMemberById">아이디</option>
15     <option value="selectMemberByPwd">암호</option>
16   </select>
17   <input type="submit" value="검색">
18 </body>
19 </html>
```

```
memberInfo.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"
3   isELIgnored="false" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%
6   request.setCharacterEncoding("UTF-8");
7 %>
8 <html>
9 <head>
10 <meta charset=UTF-8>
11 <title>회원 정보 출력창</title>
12 </head>
13 <body>
14 <table border="1" align="center" width="80%">
15   <tr align="center" bgcolor="lightgreen">
16     <td><b>아이디</b></td>
17     <td><b>비밀번호</b></td>
18     <td><b>이름</b></td>
19     <td><b>이메일</b></td>
20     <td><b>가입일</b></td>
21   </tr>
22   <tr align="center">
23     <td>${member.id}</td>
24     <td>${member.pwd}</td>
25     <td>${member.name}</td>
26     <td>${member.email}</td>
27     <td>${member.joinDate}</td>
28   </tr>
29 </table>
30 </html>
```

회원 검색창

localhost:8080/pro08/test04/search.do

입력 : 전체 검색

회원 정보 출력창

localhost:8080/pro08/test04/mem1.do?value=&action=listMembers

아이디	비밀번호	이름	이메일	가입일
ddd	111	ddd	ddd@test.com	2023-09-10
ccc	111	ccc	ccc@test.com	2023-09-10
bbb	111	111	bbb@test.com	2023-09-10
aaa	1111	aaa	aaa@test.com	2023-09-10
kim	1234	김유신	kim@test.com	2023-09-10
lee	1234	이순신	lee@test.com	2023-09-10
hong	1234	홍길동	hong@test.com	2023-09-10

회원 검색창

localhost:8080/pro08/test04/search.do

입력 : hong 아이디 검색

회원 정보 출력창

localhost:8080/pro08/test04/mem1.do?value=hong&action=selectMemberById

아이디	비밀번호	이름	이메일	가입일
hong	1234	홍길동	hong@test.com	2023-09-10

회원 검색창

localhost:8080/pro08/test04/search.do

입력 : 1234 암호 검색

회원 정보 출력창

localhost:8080/pro08/test04/mem1.do?value=1234&action=selectMemberByPwd

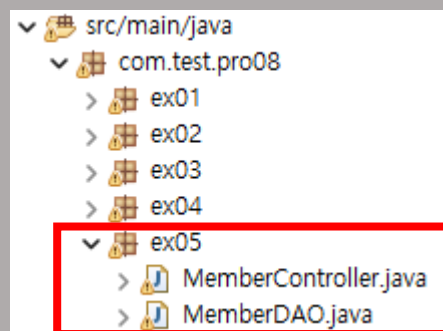
아이디	비밀번호	이름	이메일	가입일
hong	1234	홍길동	hong@test.com	2023-09-10
lee	1234	이순신	lee@test.com	2023-09-10
kim	1234	김유신	kim@test.com	2023-09-10

Quiz

✓ quiz01 패키지를 만들고 진행 하세요.


- id 로 검색해서 이름을 출력해보세요.
- id와 pwd 를 입력하고 맞으면 “로그인 성공” 틀리면 “로그인 실패”를 출력해 보세요.

✓ 회원 정보 추가 하기

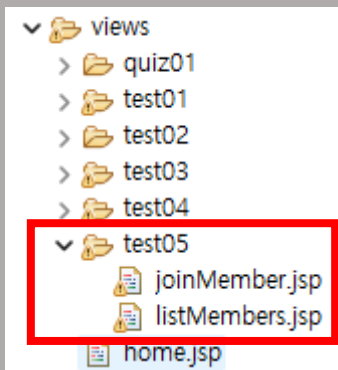


```
member.xml
44      ]]>
45    </select>
46    <select id="selectMemberByPwd" resultMap="memResult" parameterType="int">
47      <![CDATA[
48        select * from t_member where pwd=#{pwd}
49      ]]>
50    </select>
51    <insert id="insertMember" parameterType="memberVO">
52      <![CDATA[
53        insert into t_member values("#{id}, #{pwd}, #{name}, #{email}, sysdate)
54      ]]>
55    </insert>
56  </mapper>
```

```
MemberController.java
1 package com.test.pro08.ex05;
2
3 import java.util.List;
4
14 @Controller
15 @RequestMapping("/test05")
16 public class MemberController {
17     @Autowired
18     MemberDAO dao;
19
20
21     @RequestMapping(value="/insertMember.do")
22     public ModelAndView insertMember(HttpServletRequest request,
23         HttpServletResponse response) throws Exception{
24         MemberVO vo = new MemberVO();
25
26         vo.setId(request.getParameter("id"));
27         vo.setPwd(request.getParameter("pwd"));
28         vo.setName(request.getParameter("name"));
29         vo.setEmail(request.getParameter("email"));
30
31         dao.insertMember(vo);
32         ModelAndView mav = new ModelAndView("redirect:/test05/listMembers.do");
33         return mav;
34     }
35     @RequestMapping(value="/joinMember.do")
36     public ModelAndView search(HttpServletRequest request,
37         HttpServletResponse response) throws Exception{
38         ModelAndView mav = new ModelAndView("test05/joinMember");
39         return mav;
40     }
41     @RequestMapping(value="/listMembers.do")
42     public ModelAndView listMembers(HttpServletRequest request,
43         HttpServletResponse response) throws Exception{
44         List<MemberVO> membersList = dao.listMembers();
45
46         ModelAndView mav = new ModelAndView("test05/listMembers");
47         mav.addObject("membersList", membersList);
48         return mav;
49     }
50 }
```

```
MemberDAO.java ✖
1 package com.test.pro08.ex05;
2
3 import java.util.List;
4
10
11 @Repository
12 public class MemberDAO {
13     @Autowired
14     private SqlSession sqlSession;
15
16     public List<MemberVO> listMembers() {
17         List<MemberVO> membersList =
18             sqlSession.selectList("mapper.member.selectAllMemberList");
19         return membersList;
20     }
21     public void insertMember(MemberVO vo) {
22         // TODO Auto-generated method stub
23         sqlSession.insert("mapper.member.insertMember", vo);
24     }
25 }
```



```
*joinMember.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"
3   isELIgnored="false" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%
6   request.setCharacterEncoding("UTF-8");
7 %>
8 <html>
9 <head>
10 <meta charset=UTF-8>
11 <title>회원 정보 출력창</title>
12 </head>
13 <body>
14 <form action="/pro08/test05/insertMember.do">
15 <table border="1" align="center" width="80%">
16   <tr align="center" bgcolor="lightgreen">
17     <td><b>아이디</b></td>
18     <td><b>비밀번호</b></td>
19     <td><b>이름</b></td>
20     <td><b>이메일</b></td>
21   </tr>
22   <tr align="center">
23     <td><input type="text" name="id"></td>
24     <td><input type="password" name="pwd"></td>
25     <td><input type="text" name="name"></td>
26     <td><input type="text" name="email"></td>
27   </tr>
28   <tr align="center">
29     <td colspan="4">
30       <input type="submit" value="회원가입">
31       <input type="reset" value="다시입력">
32     </td>
33   </tr>
34 </table>
35 </form>
36 </html>
```

회원 정보 출력창

localhost:8080/pro08/test05/joinMember.do

아이디	비밀번호	이름	이메일
kkk	kkk	kkk@test.com
<button>회원가입</button>		<button>다시입력</button>	

회원 정보 출력창

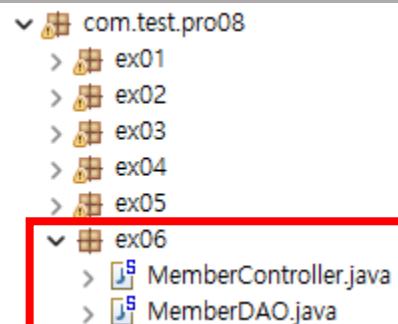
localhost:8080/pro08/test05/listMembers.do

아이디	비밀번호	이름	이메일	가입일
kkk	1234	kkk	kkk@test.com	2023-09-13
ddd	111	ddd	ddd@test.com	2023-09-10
ccc	111	ccc	ccc@test.com	2023-09-10
bbb	111	111	bbb@test.com	2023-09-10
aaa	1111	aaa	aaa@test.com	2023-09-10
kim	1234	김유신	kim@test.com	2023-09-10
lee	1234	이순신	lee@test.com	2023-09-10
hong	1234	홍길동	hong@test.com	2023-09-10


Quiz

- ✓ quiz02 패키지를 만들어서 구현 해 보세요.
 - HashMap 을 이용해서 회원 가입 해보세요.
 - 사용 방법은 거의 동일하고 단지 HashMap 이라는 차이만 존재 합니다.

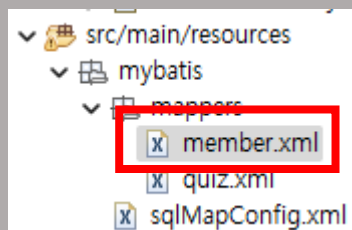
✓ 회원 정보 수정하기



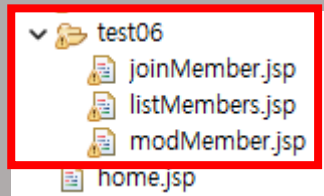
```
MemberController.java
48     mav.addObject("membersList", membersList);
49     return mav;
50 }
51 @RequestMapping(value="/updateMember")
52 public ModelAndView updateMember(@ModelAttribute("member") MemberVO vo,
53     HttpServletRequest request,
54     HttpServletResponse response) throws Exception{
55     dao.updateMember(vo);
56     ModelAndView mav = new ModelAndView("redirect:/test06/listMembers.do");
57     return mav;
58 }
59 @RequestMapping(value="/modMember.do")
60 public ModelAndView modMember(HttpServletRequest request,
61     HttpServletResponse response) throws Exception{
62     ModelAndView mav = new ModelAndView("test06/modMember");
63     return mav;
64 }
65 }
```



```
MemberDAO.java
1 package com.test.pro08.ex06;
2
3 import java.util.List;
10
11 @Repository
12 public class MemberDAO {
13     @Autowired
14     private SqlSession sqlSession;
15
16     public List<MemberVO> listMembers() {
17         List<MemberVO> membersList =
18             sqlSession.selectList("mapper.member.selectAllMemberList");
19         return membersList;
20     }
21     public void insertMember(MemberVO vo) {
22         // TODO Auto-generated method stub
23         sqlSession.insert("mapper.member.insertMember", vo);
24     }
25     public void updateMember(MemberVO vo) {
26         // TODO Auto-generated method stub
27         sqlSession.update("mapper.member.updateMember", vo);
28     }
29 }
```



```
member.xml
47      <![CDATA[
48          select * from t_member where pwd=#{pwd}
49      ]]>
50  </select>
51  <insert id="insertMember" parameterType="memberVO">
52      <![CDATA[
53          insert into t_member(id, pwd, name, email)
54              values("#{id}", #{pwd}, #{name}, #{email})
55      ]]>
56  </insert>
57  <update id="updateMember" parameterType="memberVO">
58      <![CDATA[
59          update t_member set pwd=#{pwd}, name=#{name}, email=#{email} where id=#{id}
60      ]]>
61  </update>
62 </mapper>
```



```
modMember.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"
3   isELIgnored="false" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%
6   request.setCharacterEncoding("UTF-8");
7 %>
8 <html>
9 <head>
10 <meta charset="UTF-8">
11 <title>회원 정보 수정창</title>
12 </head>
13 <body>
14 <form action="/pro08/test06/updateMember.do">
15 <table border="1" align="center" width="80%">
16   <tr align="center" bgcolor="lightgreen">
17     <td><b>아이디</b></td>
18     <td><b>비밀번호</b></td>
19     <td><b>이름</b></td>
20     <td><b>이메일</b></td>
21   </tr>
22   <tr align="center">
23     <td><input type="text" name="id"></td>
24     <td><input type="password" name="pwd"></td>
25     <td><input type="name" name="name"></td>
26     <td><input type="email" name="email"></td>
27   </tr>
28   <tr align="center">
29     <td colspan="4">
30       <input type="submit" value="수정하기">
31       <input type="reset" value="다시입력">
32     </td>
33   </tr>
34 </table>
35 </form>
36 </html>
```


회원 정보 수정창

localhost:8080/pro08/test06/modMember.do

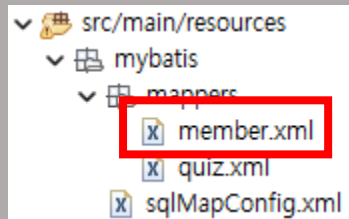
아이디	비밀번호	이름	이메일
<input type="text" value="hong"/>	<input type="password" value="...."/>	<input type="text" value="홍길순"/>	<input type="text" value="hong1@test.com"/>
<input type="button" value="수정하기"/>		<input type="button" value="다시입력"/>	

회원 정보 출력창

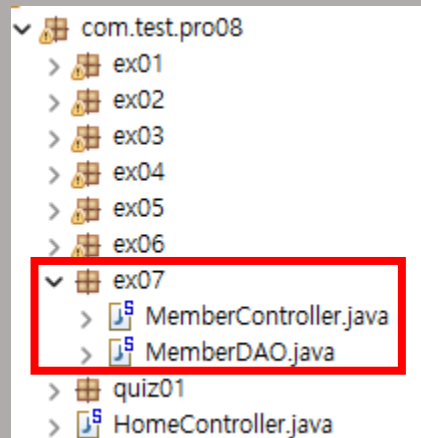
localhost:8080/pro08/test06/listMembers.do

아이디	비밀번호	이름	이메일	가입일
zzz	zzz	zzz	zzz@test.com	2023-09-10
eee	eee	eeee	eee@test.com	2023-09-10
ddd	111	ddd	ddd@test.com	2023-09-10
ccc	111	ccc	ccc@test.com	2023-09-10
bbb	111	111	bbb@test.com	2023-09-10
aaa	1111	aaa	aaa@test.com	2023-09-10
kim	1234	김유신	kim@test.com	2023-09-10
lee	1234	이수신	lee@test.com	2023-09-10
hong	4444	홍길순	hong1@test.com	2023-09-10


✓ 회원 정보 삭제 하기



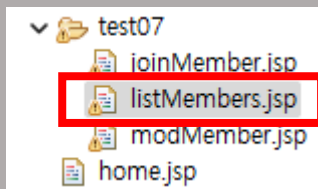
```
member.xml
52      <![CDATA[
53          insert into t_member(id, pwd, name, email)
54              values("#{id}, #{pwd}, #{name}, #{email})
55      ]]>
56  </insert>
57  <update id="updateMember" parameterType="memberVO">
58      <![CDATA[
59          update t_member set pwd=#{pwd}, name=#{name}, email=#{email} where id=#{id}
60      ]]>
61  </update>
62  <delete id="deleteMember" parameterType="String">
63      <![CDATA[
64          delete from t_member where id=#{id}
65      ]]>
66  </delete>
67 </mapper>
```



```
MemberController.java 88
57     ModelAndView mav = new ModelAndView("redirect:/test07/listMembers.do");
58     return mav;
59 }
60 @RequestMapping(value="/modMember.do")
61 public ModelAndView modMember(HttpServletRequest request,
62     HttpServletResponse response) throws Exception{
63     ModelAndView mav = new ModelAndView("test07/modMember");
64     return mav;
65 }
66 @RequestMapping(value="/deleteMember.do")
67 public ModelAndView modMember(@RequestParam String id,
68     HttpServletRequest request,
69     HttpServletResponse response) throws Exception{
70     dao.deleteMember(id);
71     ModelAndView mav = new ModelAndView("redirect:/test07/listMembers.do");
72     return mav;
73 }
74 }
```



```
MemberDAO.java
1 package com.test.pro08.ex07;
2
3 import java.util.List;
4
10
11 @Repository
12 public class MemberDAO {
13     @Autowired
14     private SqlSession sqlSession;
15
16     public List<MemberVO> listMembers() {
17         List<MemberVO> membersList =
18             sqlSession.selectList("mapper.member.selectAllMemberList");
19         return membersList;
20     }
21     public void insertMember(MemberVO vo) {
22         // TODO Auto-generated method stub
23         sqlSession.insert("mapper.member.insertMember", vo);
24     }
25     public void updateMember(MemberVO vo) {
26         // TODO Auto-generated method stub
27         sqlSession.update("mapper.member.updateMember", vo);
28     }
29     public void deleteMember(String id) {
30         // TODO Auto-generated method stub
31         sqlSession.delete("mapper.member.deleteMember", id);
32     }
33 }
```

```
*listMembers.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"
3   isELIgnored="false" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%
6   request.setCharacterEncoding("UTF-8");
7 %>
8 <html>
9 <head>
10 <meta charset=UTF-8">
11 <title>회원 정보 출력창</title>
12 </head>
13 <body>
14 <table border="1" align="center" width="80%">
15   <tr align="center" bgcolor="lightgreen">
16     <td><b>아이디</b></td>
17     <td><b>비밀번호</b></td>
18     <td><b>이름</b></td>
19     <td><b>이메일</b></td>
20     <td><b>가입일</b></td>
21     <td><b>삭제하기</b></td>
22   </tr>
23   <c:forEach var="member" items="{membersList}" >
24     <tr align="center">
25       <td>${member.id}</td>
26       <td>${member.pwd}</td>
27       <td>${member.name}</td>
28       <td>${member.email}</td>
29       <td>${member.joinDate}</td>
30       <td><a href="/pro08/test07/deleteMember.do?id=${member.id}">삭제 하기</a>
31     </td>
32   </c:forEach>
33 </table>
34 </html>
```


회원 정보 출력창

localhost:8080/pro08/test07/listMembers.do

아이디	비밀번호	이름	이메일	가입일	삭제하기
eee	eee	eeee	eee@test.com	2023-09-10	삭제 하기
ddd	111	ddd	ddd@test.com	2023-09-10	삭제 하기
ccc	111	ccc	ccc@test.com	2023-09-10	삭제 하기
bbb	111	111	bbb@test.com	2023-09-10	삭제 하기
aaa	1111	aaa	aaa@test.com	2023-09-10	삭제 하기
kim	1234	김유신	kim@test.com	2023-09-10	삭제 하기
lee	1234	이순신	lee@test.com	2023-09-10	삭제 하기
hong	1234	홍길동	hong@test.com	2023-09-10	삭제 하기

회원 정보 출력창

localhost:8080/pro08/test07/listMembers.do

아이디	비밀번호	이름	이메일	가입일	삭제하기
ddd	111	ddd	ddd@test.com	2023-09-10	삭제 하기
ccc	111	ccc	ccc@test.com	2023-09-10	삭제 하기
bbb	111	111	bbb@test.com	2023-09-10	삭제 하기
aaa	1111	aaa	aaa@test.com	2023-09-10	삭제 하기
kim	1234	김유신	kim@test.com	2023-09-10	삭제 하기
lee	1234	이순신	lee@test.com	2023-09-10	삭제 하기
hong	1234	홍길동	hong@test.com	2023-09-10	삭제 하기

Quiz

✓ 아래와 같이 구현 해 보세요.

- id 로 검색 해서 한 사람의 데이터를 출력해 보세요.
- id와 pwd로 검색해서 한 사람의 데이터를 출력해 보세요.

4. 마이바티스의 동적 SQL문 사용하기

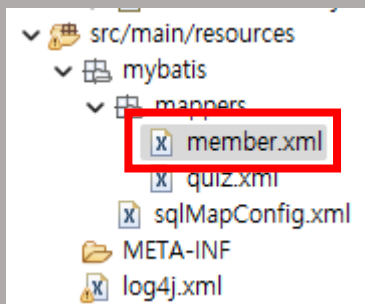
- ✓ 공통 SQL문에 대해 조건값의 유무에 따라 동적으로 처리 하기 위해서 사용하며 조건절을 추가 할 수 있다.
- ✓ 마이바티스의 동적 SQL문의 특징
 - 주로 SQL문의 조건절에서 사용한다.
 - 조건절(where)에 조건을 동적으로 추가할 수 있다.
 - JSTL과 XML 기반으로 동적 SQL문을 작성할 수 있다.

✓ 마이바티스의 동적 SQL문 구성 요소

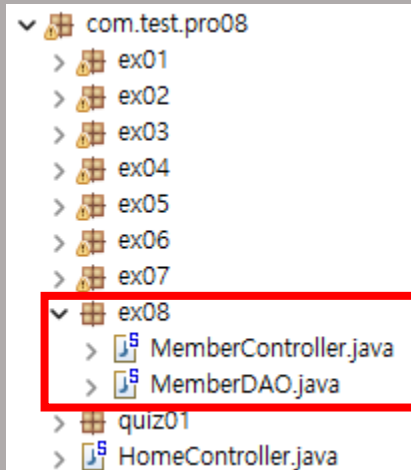
- if
- choose(when, otherwise)
- trim(where, set)
- foreach

✓ <if> 태그로 동적 SQL 문 만들기


```
<where>  
    <if test="조건식">  
        추가할 구문  
    </if>  
</where>
```



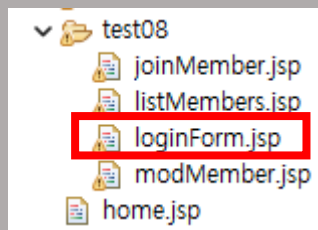
```
member.xml
63      <![CDATA[
64          delete from t_member where id=#{id}
65      ]]>
66  </delete>
67  <select id="searchMember" parameterType="memberVO" resultMap="memResult">
68      <![CDATA[
69          select * from t_member
70      ]]>
71  <where>
72      <if test="id != '' and id != null">
73          id = #{id}
74      </if>
75      <if test="pwd != '' and pwd != null">
76          and pwd = #{pwd}
77      </if>
78  </where>
79      order by joinDate desc
80  </select>
81 </mapper>
```

```
MemberController.java 88
73     }
74     @RequestMapping(value="/searchMember.do")
75     public ModelAndView searchMember(@ModelAttribute("vo") MemberVO vo,
76         HttpServletRequest request,
77         HttpServletResponse response) throws Exception{
78
79         List<MemberVO> membersList =dao.searchMember(vo);
80         ModelAndView mav = new ModelAndView("test08/listMembers");
81         mav.addObject("membersList",membersList);
82         return mav;
83     }
84     @RequestMapping(value="/loginForm.do")
85     public ModelAndView loginForm(HttpServletRequest request,
86         HttpServletResponse response) throws Exception{
87         ModelAndView mav = new ModelAndView("test08/loginForm");
88         return mav;
89     }
90 }
```



```
*MemberDAO.java  ✖
25 public void updateMember(MemberVO vo) {
26     // TODO Auto-generated method stub
27     sqlSession.update("mapper.member.updateMember", vo);
28 }
29 public void deleteMember(String id) {
30     // TODO Auto-generated method stub
31     sqlSession.delete("mapper.member.deleteMember", id);
32 }
33 public List<MemberVO> searchMember(MemberVO vo) {
34     // TODO Auto-generated method stub
35     List<MemberVO> memberList = sqlSession.selectList("mapper.member.searchMember", vo);
36     return memberList;
37 }
38 }
```



```
*loginForm.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"
3     isELIgnored="false" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%
6     request.setCharacterEncoding("UTF-8");
7 %>
8 <html>
9 <head>
10 <meta charset=UTF-8">
11 <title>로그인</title>
12 </head>
13 <body>
14 <form action="/pro08/test08/searchMember.do">
15 <table border="1" align="center" width="80%">
16     <tr align="center" bgcolor="lightgreen">
17         <td ><b>아이디</b></td>
18         <td><b>비밀번호</b></td>
19     </tr>
20     <tr align="center">
21         <td><input type="text" name="id"></td>
22         <td><input type="password" name="pwd"></td>
23     </tr>
24     <tr align="center">
25         <td colspan="4">
26             <input type="submit" value="로그인">
27             <input type="reset" value="다시입력">
28         </td>
29     </tr>
30 </table>
31 </form>
32 </html>
```

로그인

localhost:8080/pro08/test08/loginForm.do

시크릿 모드(창 2개)

아이디	비밀번호
<input type="text" value="hong"/>	<input type="password"/>
<div>로그인</div> <div>다시입력</div>	

회원 정보 출력창

localhost:8080/pro08/test08/searchMember.do?id=hong&pwd=

시크릿 모드(창 2개)

아이디	비밀번호	이름	이메일	가입일	삭제하기
hong	1234	홍길동	hong@test.com	2023-09-10	삭제하기

로그인

localhost:8080/pro08/test08/loginForm.do

시크릿 모드(창 2개)

아이디	비밀번호
<input type="text" value="lee"/>	<input type="password" value="...."/>
<div>로그인</div> <div>다시입력</div>	

회원 정보 출력창

localhost:8080/pro08/test08/searchMember.do?id=lee&pwd=1234

시크릿 모드(창 2개)

아이디	비밀번호	이름	이메일	가입일	삭제하기
lee	1234	이순신	lee@test.com	2023-09-10	삭제하기

로그인

localhost:8080/pro08/test08/loginForm.do

아이디	비밀번호
<input type="text"/>	<input type="password"/>
<input type="button" value="로그인"/> <input type="button" value="다시입력"/>	

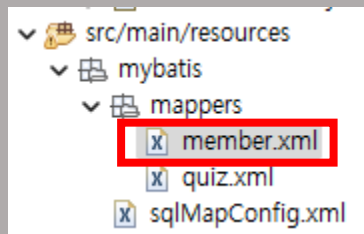
회원 정보 출력창

localhost:8080/pro08/test08/searchMember.do?id=&pwd=

아이디	비밀번호	이름	이메일	가입일	삭제하기
ddd	111	ddd	ddd@test.com	2023-09-10	삭제 하기
ccc	111	ccc	ccc@test.com	2023-09-10	삭제 하기
bbb	111	111	bbb@test.com	2023-09-10	삭제 하기
aaa	1111	aaa	aaa@test.com	2023-09-10	삭제 하기
kim	1234	김유신	kim@test.com	2023-09-10	삭제 하기
lee	1234	이순신	lee@test.com	2023-09-10	삭제 하기
hong	1234	홍길동	hong@test.com	2023-09-10	삭제 하기

✓ <choose> 태그로 동적 SQL문 만들기

```
<where>  
  <when test="조건식1">  
    구문1  
  </when>  
  
  <when test="조건식2">  
    구문2  
  </when>  
  <otherwise>  
    구문 n+1  
  </otherwise>  
</where>
```

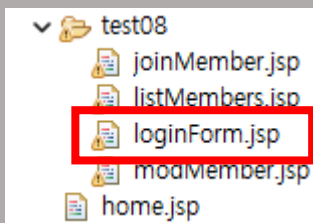


```
member.xml 88
78         </where>
79         order by joinDate desc
80     </select>
81     <select id="searchMember2" parameterType="memberVO" resultMap="memResult">
82         <![CDATA[
83             select * from t_member
84         ]]>
85     <where>
86         <choose>
87             <when test="id != '' and id != null and pwd != null and pwd != ''">
88                 id=#{id} and pwd=#{pwd}
89             </when>
90             <when test="id != '' and id != null">
91                 id=#{id}
92             </when>
93             <when test="pwd != '' and pwd != null">
94                 pwd=#{pwd}
95             </when>
96         </choose>
97     </where>
98     order by joinDate desc
99 </select>
100 </mapper>
```

> MemberController.java
> MemberDAO.java

```
MemberController.java 82         return mav;  
83     }  
84     @RequestMapping(value="/loginForm.do")  
85     public ModelAndView loginForm(HttpServletRequest request,  
86                                 HttpServletResponse response) throws Exception{  
87         ModelAndView mav = new ModelAndView("test08/loginForm");  
88         return mav;  
89     }  
90     @RequestMapping(value="/searchMember2.do")  
91     public ModelAndView searchMember2(@ModelAttribute("vo") MemberVO vo,  
92                                     HttpServletRequest request,  
93                                     HttpServletResponse response) throws Exception{  
94  
95         List<MemberVO> membersList =dao.searchMember2(vo);  
96         ModelAndView mav = new ModelAndView("test08/listMembers");  
97         mav.addObject("membersList",membersList);  
98         return mav;  
99     }  
100 }
```

```
MemberDAO.java
31     sqlSession.delete("mapper.member.deleteMember", id);
32 }
33 public List<MemberVO> searchMember(MemberVO vo) {
34     // TODO Auto-generated method stub
35     List<MemberVO> memberList = sqlSession.selectList("mapper.member.searchMember", vo);
36     return memberList;
37 }
38 public List<MemberVO> searchMember2(MemberVO vo) {
39     // TODO Auto-generated method stub
40     List<MemberVO> memberList = sqlSession.selectList("mapper.member.searchMember2", vo);
41     return memberList;
42 }
43 }
```



```
*loginForm.jsp
10 <meta charset=UTF-8">
11 <title>로그인</title>
12 </head>
13 <body>
14 <form action="/pro08/test08/searchMember2.do">
15 <table border="1" align="center" width="80%">
16 <tr align="center" bgcolor="lightgreen">
17 <td><b>아이디</b></td>
18 <td><b>비밀번호</b></td>
19 </tr>
```

로그인

localhost:8080/pro08/test08/loginForm.do

☆ □ 시크릿 모드

아이디

비밀번호

aaa

로그인

다시입력

회원 정보 출력창

localhost:8080/pro08/test08/searchMember2.do?id=aaa&pwd=

☆ □ 시크릿 모드

아이디	비밀번호	이름	이메일	가입일	삭제하기
aaa	1111	aaa	aaa@test.com	2023-09-10	삭제 하기

로그인

localhost:8080/pro08/test08/loginForm.do

☆ □ 시크릿 모드

아이디

비밀번호

....

로그인

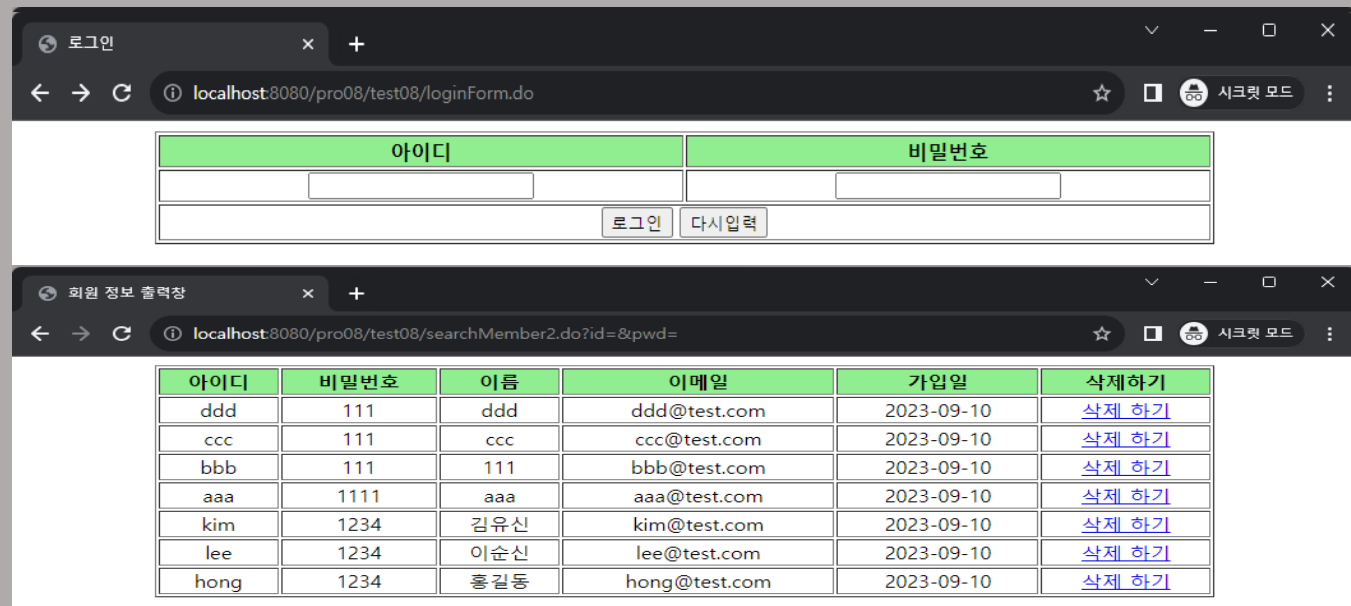
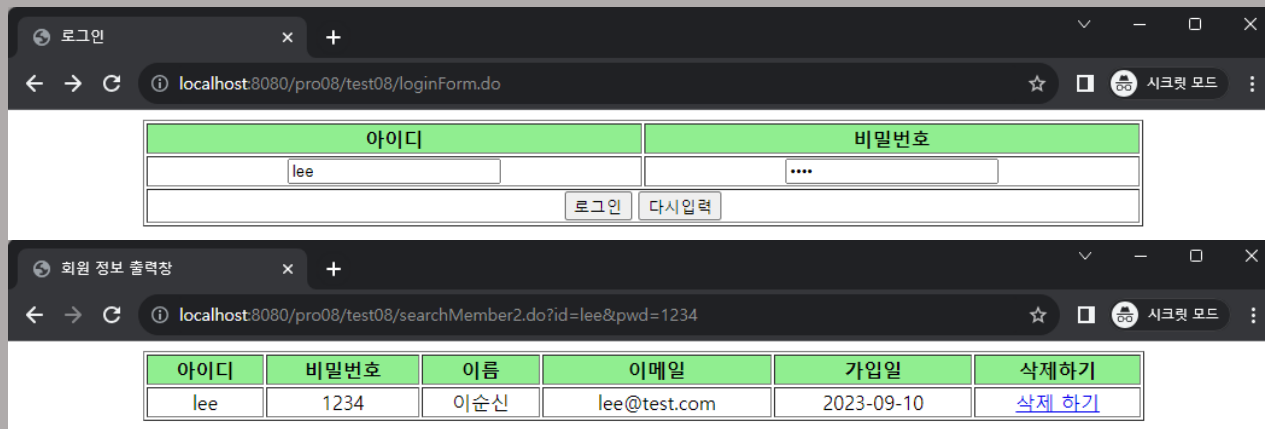
다시입력

회원 정보 출력창

localhost:8080/pro08/test08/searchMember2.do?id=&pwd=1234

☆ □ 시크릿 모드

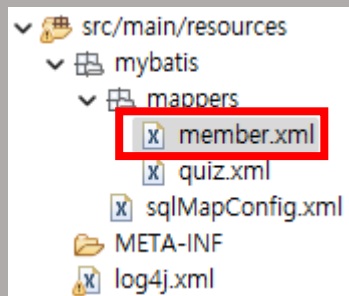
아이디	비밀번호	이름	이메일	가입일	삭제하기
kim	1234	김유신	kim@test.com	2023-09-10	삭제 하기
lee	1234	이순신	lee@test.com	2023-09-10	삭제 하기
hong	1234	홍길동	hong@test.com	2023-09-10	삭제 하기



✓ <foreach> 태그로 회원 정보 조회하기

```
<foreach item="item" collection="list" index="index" open="(" close=")" separator=",">
    #{item}
</foreach>
```

속성	설명
collection	전달받은 인자 값을 의미하며, 배열과 List 계열 인스턴스를 전달할 수 있다. List 인스턴스로 전달 할 때는 list로 표시하고 배열로 전달 할 때는 array로 표시한다.
index	foreach문이 반복될 때마다 1씩 증가시키면서 접근하는 값의 위치를 나타낸다. 최초 값의 위치는 0이다.
item	반복문이 실행 될때 마다 collection 속성에 지정된 값에 접근하여 차례대로 사용한다.
open	해당 구문이 시작될 때의 지정한 기호를 추가한다.
close	해당 구문이 끝날 때의 지정한 기호를 추가한다.
separator	한 번 이상 반복될 때 반복되는 사이에 지정한 기호를 추가한다.



```
member.xml
94         pwd=#{pwd}
95     </when>
96 </choose>
97 </where>
98     order by joinDate desc
99 </select>
100 <select id="foreachSelect" resultMap="memResult" parameterType="java.util.Map">
101     <![CDATA[
102         select * from t_member
103     ]]>
104     where name in
105     <foreach item="item" collection="list" open="(" separator="," close=")">
106         #{item}
107     </foreach>
108     order by joinDate desc
109 </select>
110 </mapper>
```

✓ ex09
> MemberController.java
> MemberDAO.java

```
MemberController.java ✕
96     List<MemberVO> membersList = dao.searchMember2(vo);
97     ModelAndView mav = new ModelAndView("test09/listMembers");
98     mav.addObject("membersList", membersList);
99     return mav;
100 }
101 @RequestMapping(value="/foreachSelect.do")
102 public ModelAndView foreachSelect(HttpServletRequest request,
103     HttpServletResponse response) throws Exception{
104     List<String> nameList = new ArrayList<String>();
105     nameList.add("홍길동");
106     nameList.add("이순신");
107     nameList.add("손흥민");
108     List membersList = dao.foreachSelect(nameList);
109     ModelAndView mav = new ModelAndView("test09/listMembers");
110     mav.addObject("membersList", membersList);
111     return mav;
112 }
113 }
```

```

MemberDAO.java
37     }
38     public List<MemberVO> searchMember2(MemberVO vo) {
39         // TODO Auto-generated method stub
40         List<MemberVO> memberList = sqlSession.selectList("mapper.member.searchMember2", vo);
41         return memberList;
42     }
43     public List<MemberVO> foreachSelect(List<String> nameList) {
44         // TODO Auto-generated method stub
45         List<MemberVO> membersList = sqlSession.selectList("mapper.member.foreachSelect", nameList);
46         return membersList;
47     }
48 }

```

회원 정보 출력창

localhost:8080/pro08/test09/foreachSelect.do

아이디	비밀번호	이름	이메일	가입일	삭제하기
lee	1234	이순신	lee@test.com	2023-09-10	삭제하기
hong	1234	홍길동	hong@test.com	2023-09-10	삭제하기

Quiz

✓ <foreach> 태그로 회원 정보를 추가해 보세요.

✓ <sql> 태그와 <include> 태그로 SQL문 중복 제거 하기

```
member.xml
67
68 <sql id="sel">
69     <![CDATA[
70         select * from t_member
71     ]]>
72 </sql>
73 <select id="searchMember" parameterType="memberVO" resultMap="memResult">
74     <!-- <![CDATA[
75         select * from t_member
76     ]]> -->
77     <include refid="sel"></include>
78     <where>
79         <if test="id != '' and id != null">
80             id = #{id}
81         </if>
82         <if test="pwd != '' and pwd != null">
83             and pwd = #{pwd}
84         </if>
85     </where>
86     order by joinDate desc
87 </select>
88 <select id="searchMember2" parameterType="memberVO" resultMap="memResult">
89     <![CDATA[
```

✓ 마이바티스에서 오라클 연동 했을 때 list 검색하는 방법

```
select * from t_member where name like '%길%';
```

ID	PWD	NAME	EMAIL	JOINDATE
1	hong1234	홍길동	hong@test.com	23/09/10

- 오라클 DB에서는 ‘%길%’로 처리하는데 마이바티스에서는 ‘%길%’를 처리 하지 못한다. 그래서 오라클 DB에서 문자열 연결 기호인 || 를 이용하여 사용한다.

```
*member.xml
117 <select id="selectLike" resultMap="memResult" parameterType="String">
118     <include refid="sel"></include>
119     where name like '%'||'길'||'%'
120 </select>
121 </mapper>
```

Quiz

✓ 이메일 검색으로 이메일이 같은 글자가 들어 있는 Member를 출력 해 보세요.

A photograph of a server room with rows of server racks on both sides of a central aisle. The racks have glass doors and are filled with server units, some of which have glowing blue indicator lights. The ceiling has several long, rectangular light fixtures. The overall atmosphere is dimly lit, with the primary light sources being the server lights and the ceiling fixtures.

수고하셨습니다.