박 경 태
comsi.java@gmail.com

고급 자바 프로그래밍
: STS를 이용한 Spring 프로그래밍

Mybatis 프레임워크 시작하기

BoardDAO 비교하기 – JDBC vs. MyBatis

```java
public class BoardDAO {
    // JDBC 관련 변수
    private Connection conn = null;
    private PreparedStatement pstmt = null;
    private ResultSet rset = null;


    private final String BOARD_LIST = "select * from board order by seq desc";

    // 글 목록 조회
    public List<BoardVO> getBoardList(BoardVO vo) {
        System.out.println("==> JDBC로 getBoardList() 기능 처리");
        List<BoardVO> boardList = new ArrayList<BoardVO>();
        try {
            conn = JDBCUtil.getConnection();
            pstmt = conn.prepareStatement(BOARD_LIST);
            rset = pstmt.executeQuery();
            while (rset.next()) {
                BoardVO board = new BoardVO();
                board.setSeq(rset.getInt("SEQ"));
                board.setTitle(rset.getString("TITLE"));
                board.setWriter(rset.getString("WRITER"));
                board.setContent(rset.getString("CONTENT"));
                board.setRegDate(rset.getDate("REGDATE"));
                board.setCnt(rset.getInt("CNT"));
                boardList.add(board);
            }
        }catch(Exception e) {
            e.printStackTrace();
        }finally {
            JDBCUtil.close(rset, pstmt, conn);
        }
        return boardList;
    }
}
```

JDBC

```java
public class BoardDAO {

    public List<BoardVO> getBoardList(BoardVO vo) {
        SqlSession mybatis = SqlSessionFactoryBean.getSqlSessionInstance();

        return mybatis.selectList("BoardDAO.getBoardList", vo);
    }
}
```

MyBatis

1.1 Mybatis 프레임워크 2가지 특징

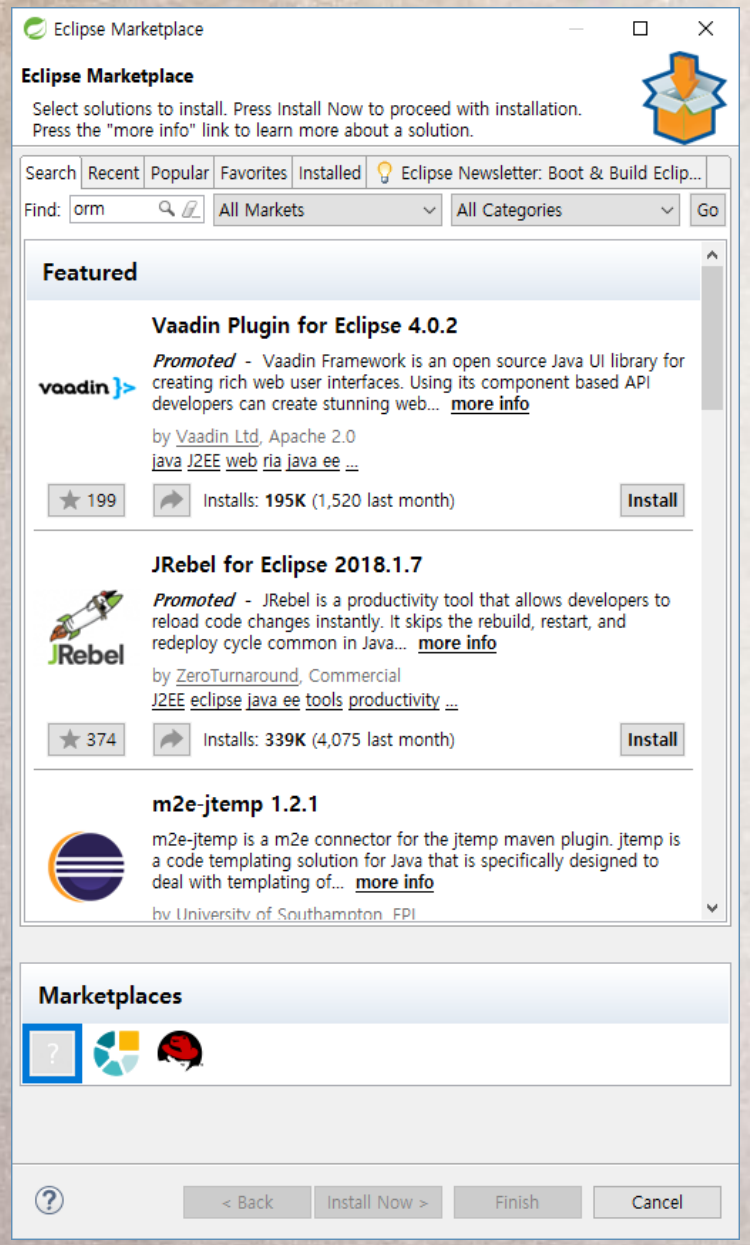
- **한 두 줄의 자바 코드로 DB 연동을 처리한다는 것**
  - XML 파일에 저장된 SQL 명령어를 대신 실행하고 실행 결과를 VO 자바 객체에 자동으로 매핑시켜 줌. ➔ 데이터맵퍼(Data Mapper)라고 함

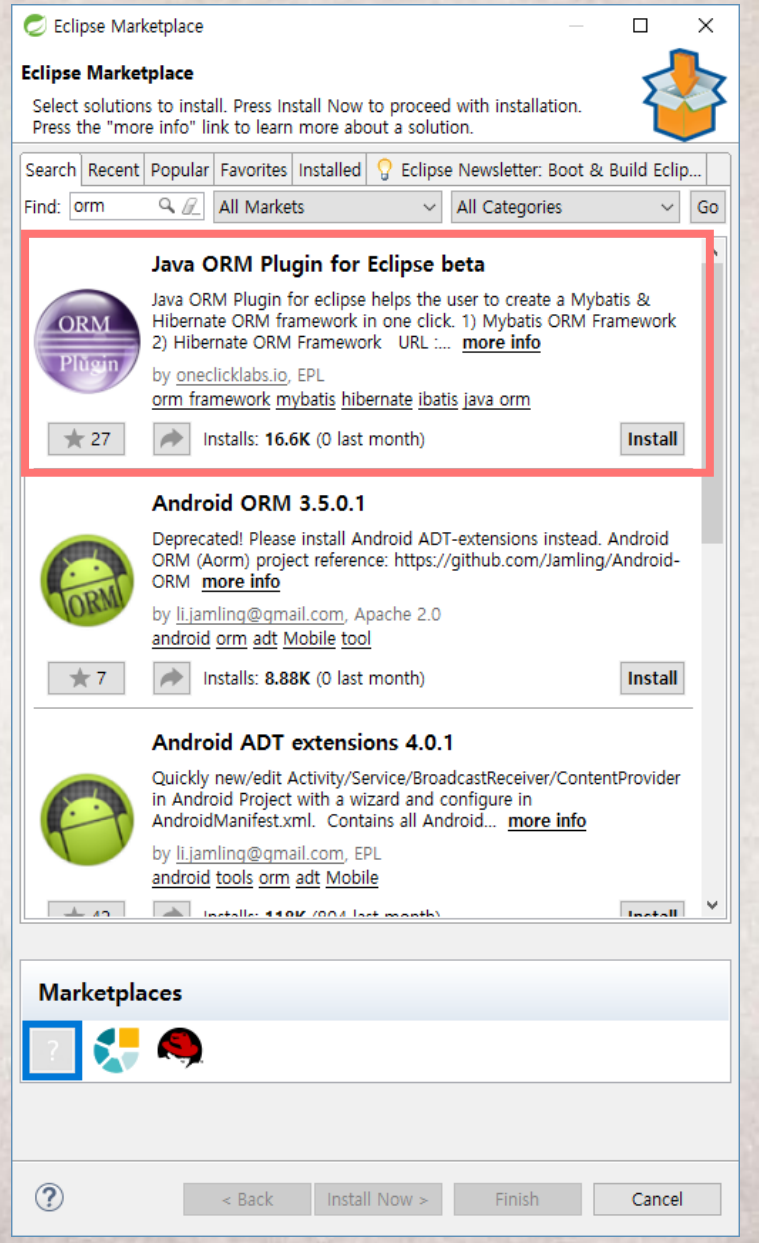
- **SQL 명령어를 자바 코드에서 분리하여 XML파일에 따로 관리한다는 것**
  - 만약, SQL 명령어가 DAO 같은 자바 클래스에 저장되면 SQL 명령어만 수정하는 상황에서도 자바 클래스를 다시 컴파일해야 한다.
  - SQL 명령어들을 한 곳에 모아서 관리하기도 쉽지 않다.
  - 결국, SQL 명령어에 대한 통합 관리를 위해서라도 자바 소스에서 SQL을 분리하는 게 매우 중요함.

XML파일 분리의 예

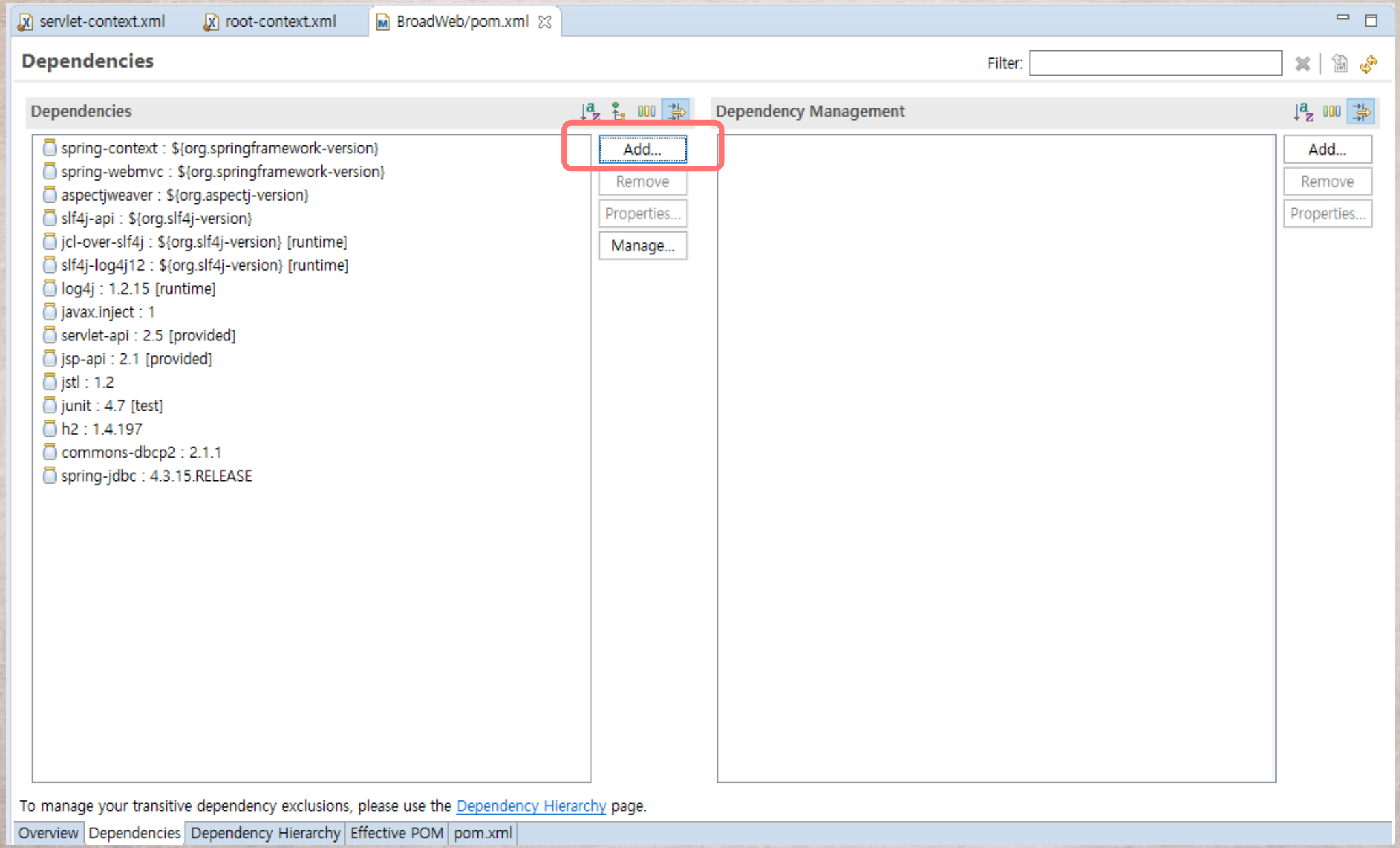
```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="kr.ac.inje.comsi.board.BoardVO">


        <select id="getBoardList" resultType="kr.ac.inje.comsi.board.BoardVO" >
            select * from board
        </select>


</mapper>
```

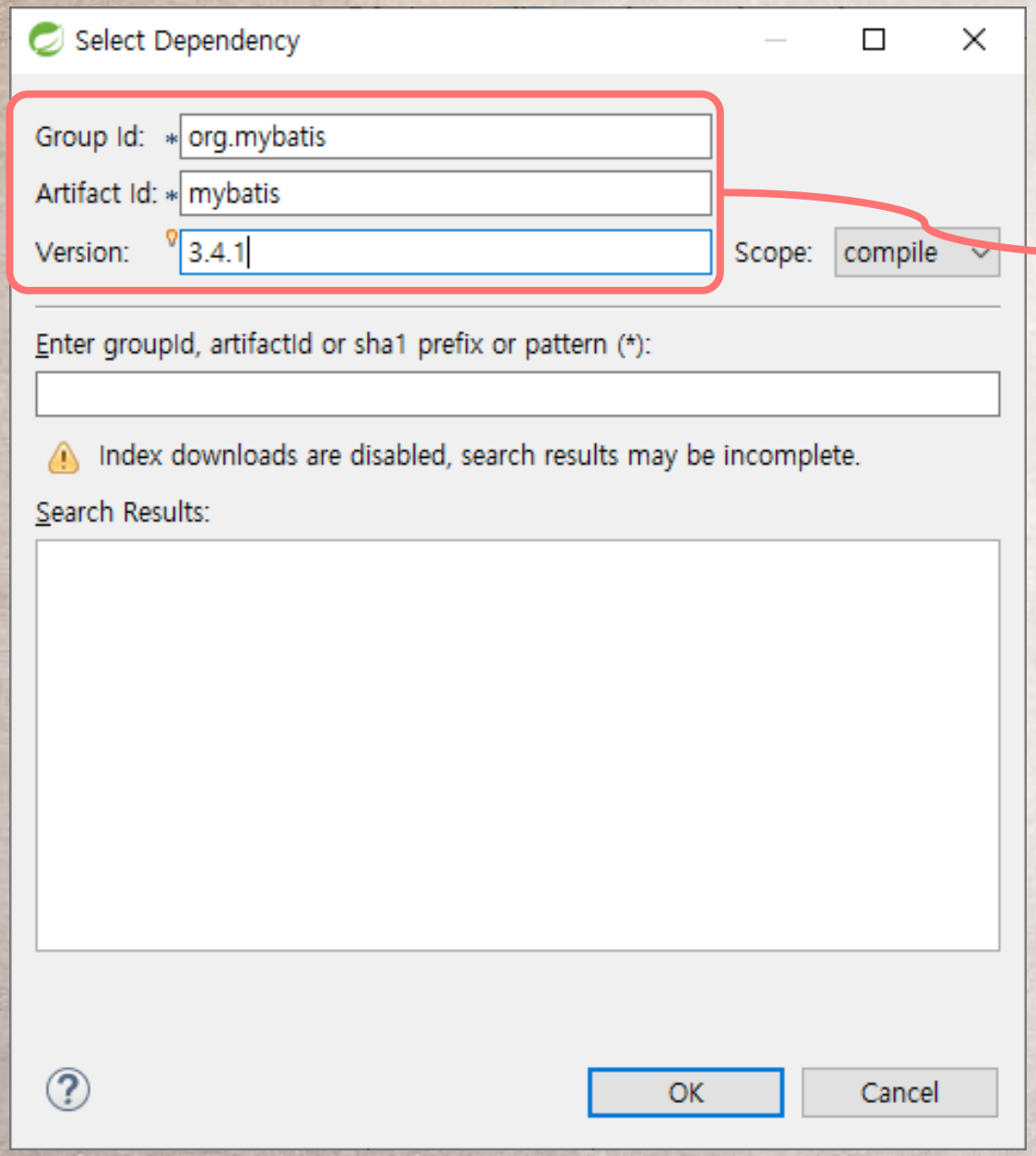
1.2 ORM 플러그인 설치→ 제대로 다운로드가 안 됨ㅠㅠ



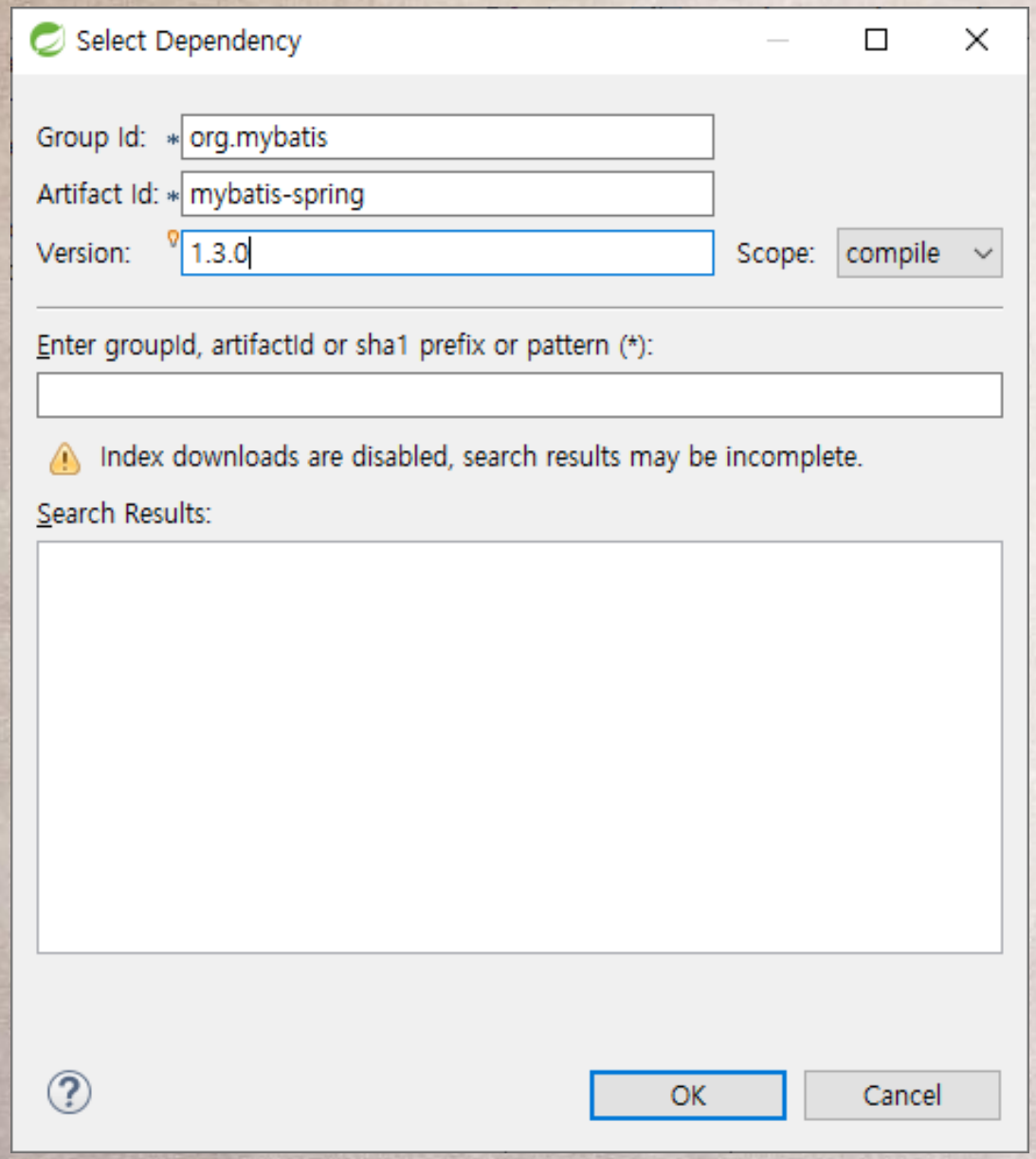


1.3 라이브러리 추가 – Maven Dependency(pom.xml파일)

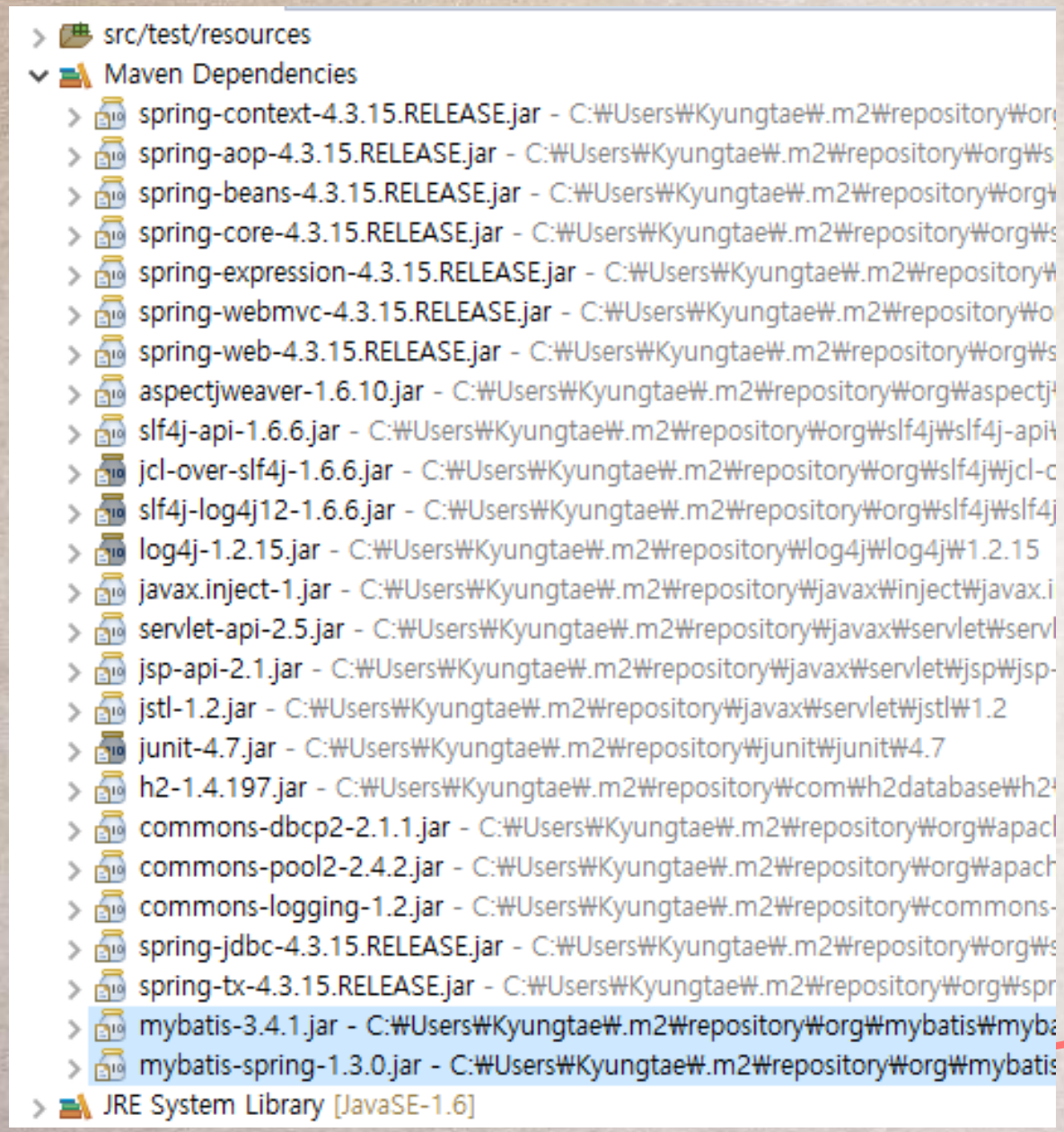




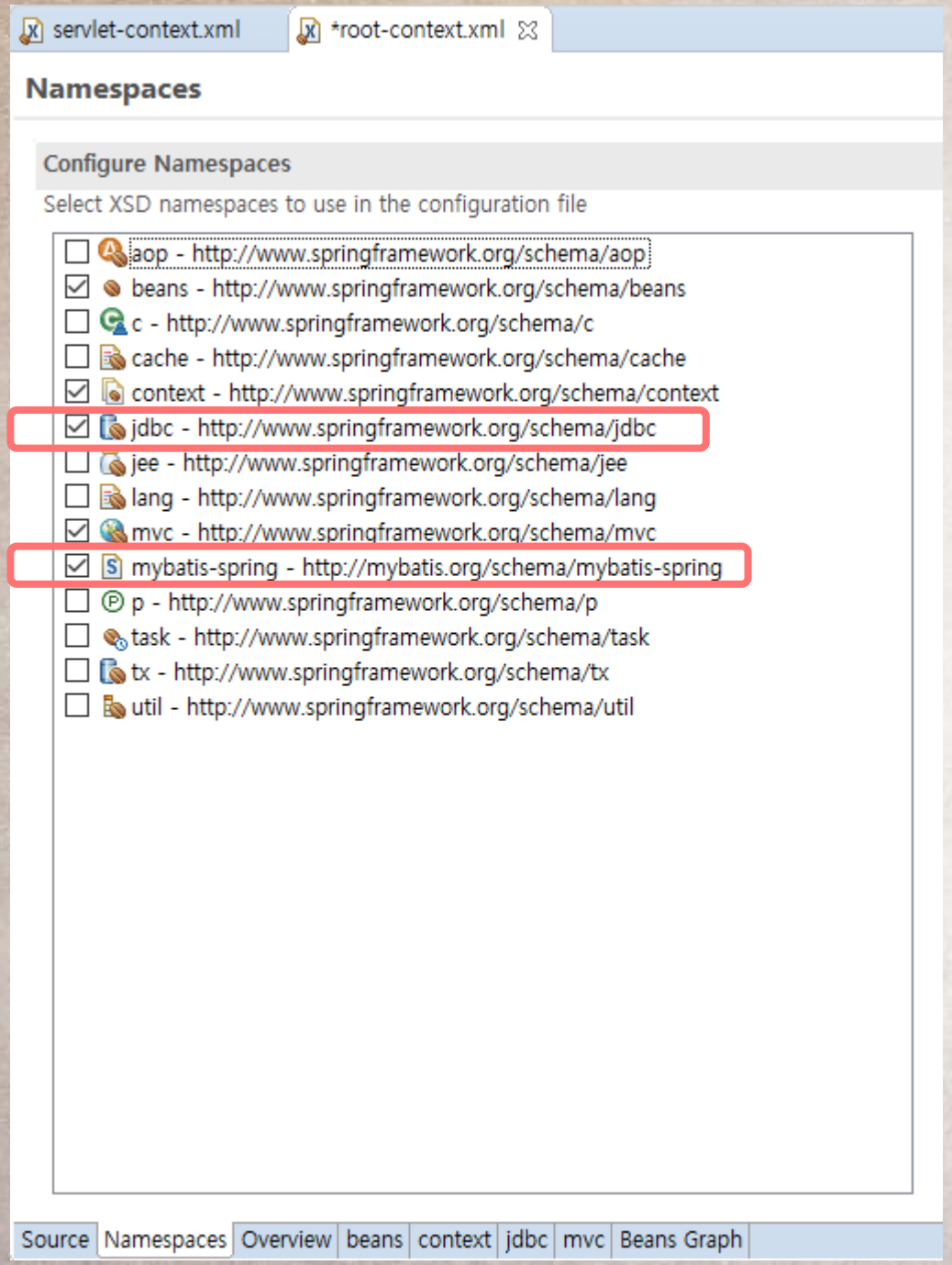
Group Id: org.mybatis
Artifact Id: mybatis
Version: 3.4.1



Group Id: org.mybatis
Artifact Id: mybatis-spring
Version: 1.3.0



mybatis-3.4.1.jar
mybatis-spring-1.3.0.jar



root-context.xml 파일

- Namespace 항목 추가하기
  - jdbc
  - mybatis-spring

1.4 root-context.xml 수정-DataSource 추석 추가

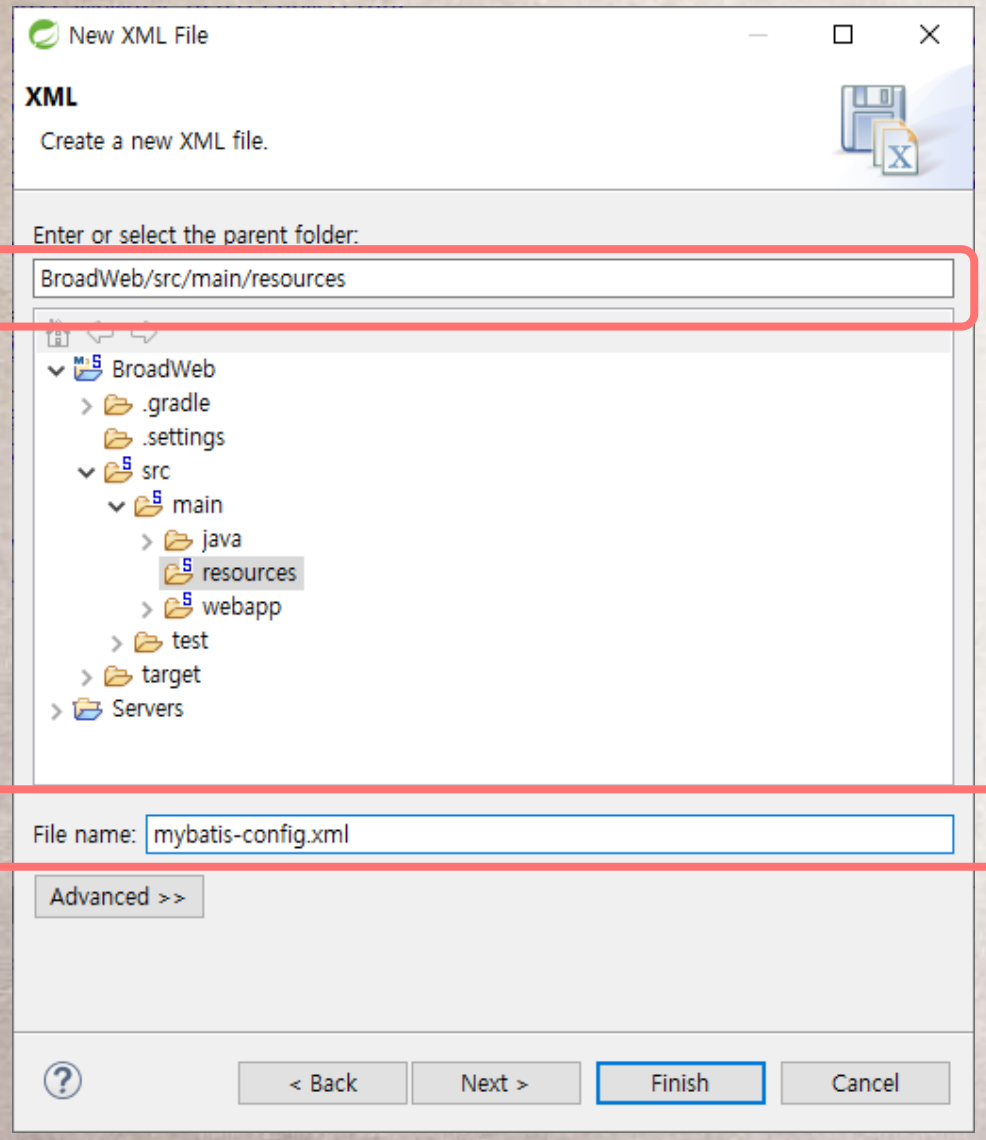
servlet-context.xml | root-context.xml | applicationContext.xml

```
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      xmlns:context="http://www.springframework.org/schema/context"
5      xmlns:mvc="http://www.springframework.org/schema/mvc"
6      xmlns:mybatis-spring="http://mybatis.org/schema/mybatis-spring"
7      xmlns:jdbc="http://www.springframework.org/schema/jdbc"
8      xsi:schemaLocation="http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-4.3.xsd
9          http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-4.3.xsd
10         http://mybatis.org/schema/mybatis-spring http://mybatis.org/schema/mybatis-spring-1.2.xsd
11         http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.3.xsd
12         http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.3.xsd">
13
14     <context:component-scan base-package="kr.ac.inje.comsi.board"></context:component-scan>
15
16     <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
17         <property name="driverClassName" value="org.h2.Driver"></property>
18         <property name="url" value="jdbc:h2:D:/workspace/h2db/db"></property>
19         <property name="username" value="sa"></property>
20         <property name="password" value=""></property>
21     </bean>
22
23 </beans>
24
25
```

주석 처리하기(삭제)

1.5 SQL Mapping 프레임워크

- MyBatis는 SQL Mapping 프레임워크로 별도의 설정파일을 가짐



BoardWeb/src/main/resources

mybatis-config.xml

mybatis-config.xml 파일 수정 – typeAlias 설정

- <typeAliases> 엘리먼트는 <typeAlias>를 여러 개 가질 수 있으며,
- <typeAlias>를 사용하여 특정 클래스의 별칭(alias)를 선언할 수 있다.
- "kr.ac.inje.comsi.board.BoardVO"클래스를 "board" 별칭으로 선언
- *-mapping.xml 파일에서 접근 사용가능

root-context.xml | mybatis-config.xml | board-mapping.xml | BoardDAO.java

```
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 alias="board" type="kr.ac.inje.comsi.board.BoardVO"/>
8      </typeAliases>
9  </configuration>
10
11
```

mybatis-config.xml 내용 파일 추가

root-context.xml | mybatis-config.xml | board-mapping.xml | BoardDAO.java | SqlSessionFactoryBean.java | BoardServiceClient.java

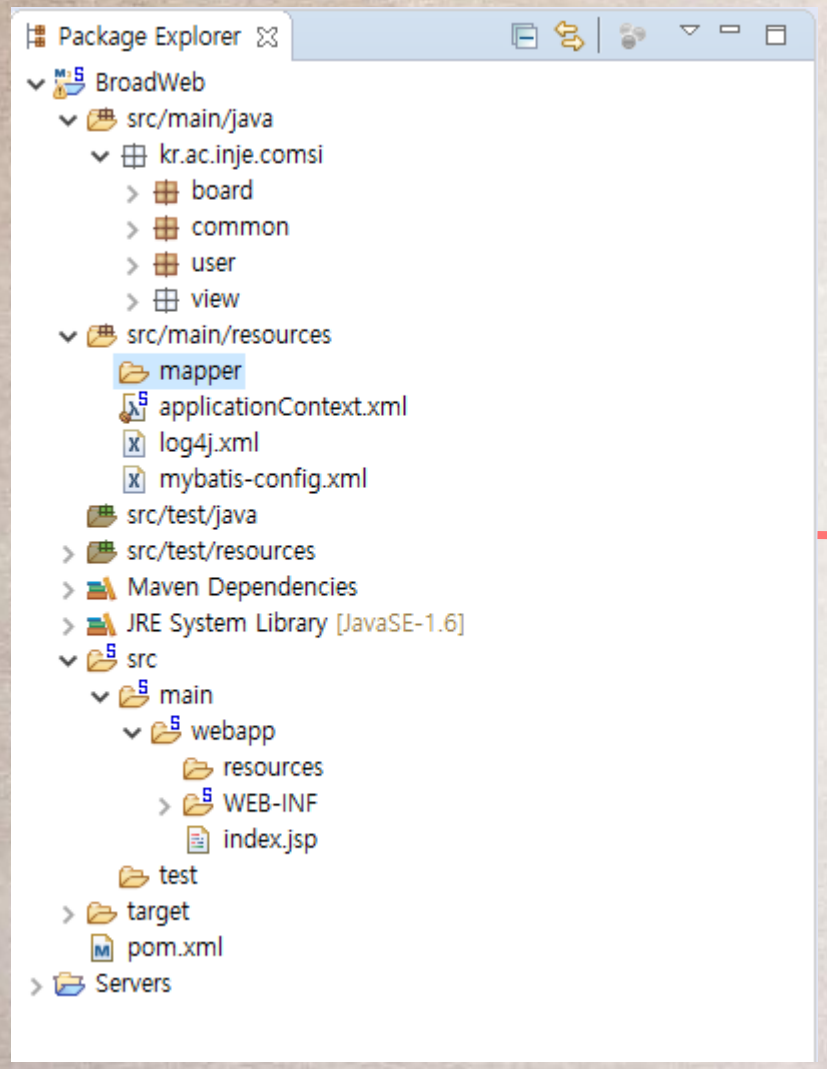
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
3                                 "http://mybatis.org/dtd/mybatis-3-config.dtd">
4
5  <configuration>
6      <!-- <properties resource="db.properties"/> -->
7      <typeAliases>
8          <typeAlias alias="board" type="kr.ac.inje.comsi.board.BoardVO"/>
9      </typeAliases>
10     <environments default="development">
11         <environment id="development">
12             <transactionManager type="JDBC"/>
13             <dataSource type="POOLED">
14                 <property name="driver" value="org.h2.Driver"></property>
15                 <property name="url" value="jdbc:h2:D:/workspace/h2db/db"></property>
16                 <property name="username" value="sa"></property>
17                 <property name="password" value=""></property>
18             </dataSource>
19         </environment>
20     </environments>
21
22     <mappers>
23         <mapper resource="mapper/board-mapping.xml"/>
24     </mappers>
25 </configuration>
26
```

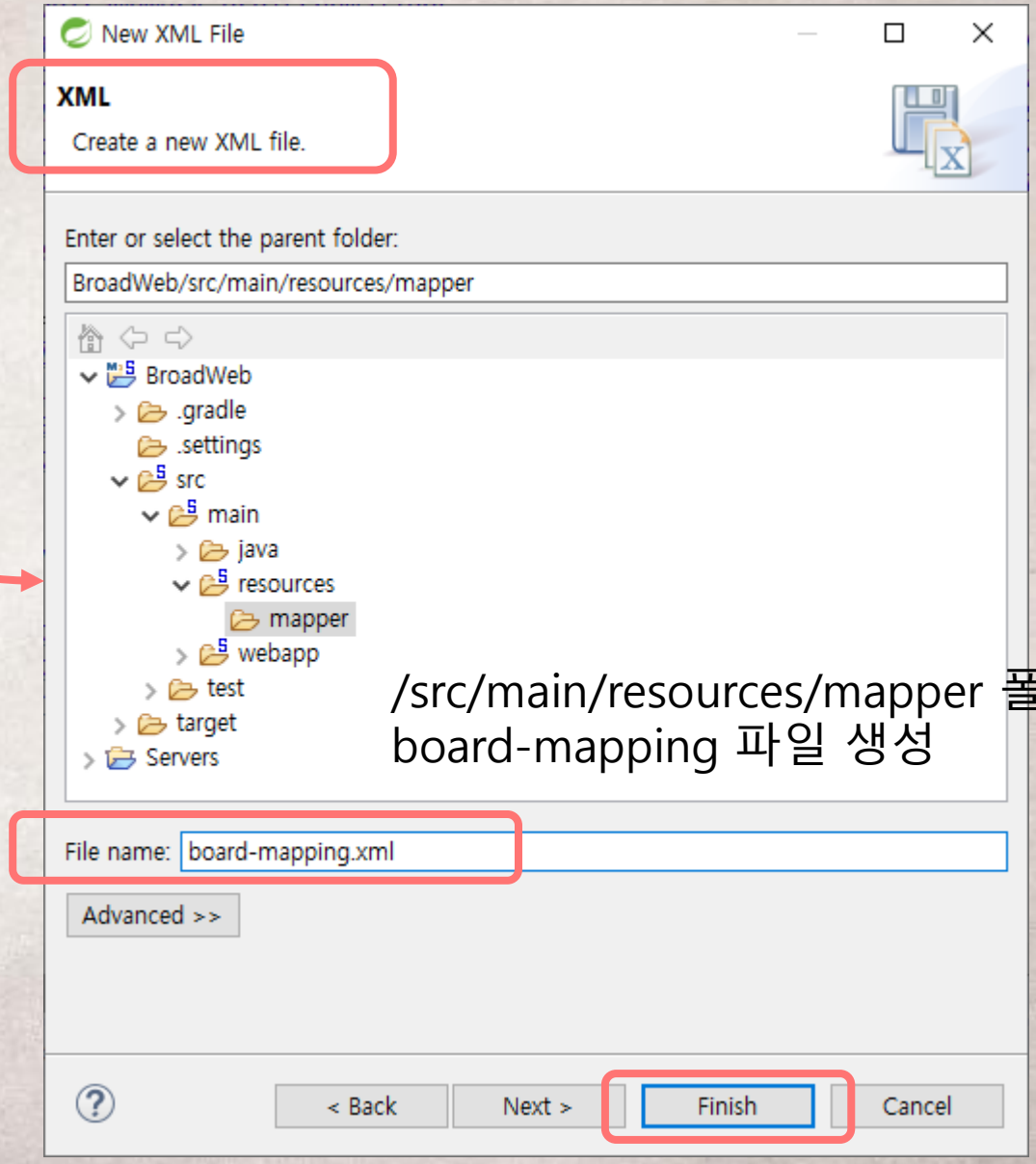
SQL mapper 파일에서 호출 사용

sql mapper 파일 위치 설정

➔ http://www.mybatis.org/mybatis-3/ko/getting-started.html

1.6 SQL Mapper 파일 설정 – Board Table





board-mapping.xml의 DTD 및 <mapper>root 엘리먼트 설정

root-context.xml | mybatis-config.xml | board-mapping.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  <mapper>
5
6  </mapper>
7
```

board-mapping.xml 내용 파일 추가

root-context.xml | mybatis-config.xml | board-mapping.xml | BoardDAO.java | SqlSessionFactoryBean.java | BoardServiceClient.java

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
                        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="BoardDAO">
    <insert id="insertBoard">
        insert into board(seq, title, writer, content)
            values((select nvl(max(seq),0)+1 from board),#{title}, #{writer},#{content})
    </insert>

    <select id="getBoard" resultType="board">
        select * from board where seq=#{seq}
    </select>

    <select id="getBoardList" resultType="board">
        select * from board order by seq desc
    </select>

    <delete id="deleteBoard">
        delete board where seq=#{seq}
    </delete>

    <update id="updateBoard">
        update board set title=#{title}, content=#{content} where seq=#{seq}
    </update>
</mapper>
```

1.7 SqlSession객체 생성하기-SqlSessionFactoryBean 클래스

- Mybatis에서 DAO를 구현할려면 SqlSession 객체가 필요
- SqlSession 객체를 얻으려면 SqlSessionFactory개체가 필요

root-context.xml | mybatis-config.xml | board-mapping.xml | BoardDAO.java | SqlSessionFactoryBean.java | BoardServiceClient.java

```java
package kr.ac.inje.comsi.board;

import java.io.Reader;

import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

public class SqlSessionFactoryBean {
    private static SqlSessionFactory sessionFactory = null;

    static {
        try {
            if (sessionFactory == null) {
                Reader reader = Resources.getResourceAsReader("mybatis-config.xml");
                sessionFactory = new SqlSessionFactoryBuilder().build(reader);
            }
        }catch(Exception e) {
            e.printStackTrace();
        }
    }

    public static SqlSession getSqlSessionInstance() {
        return sessionFactory.openSession();
    }
}
```

SqlSession객체를 넘긴다.

BoardDAO 수정

root-context.xml | mybatis-config.xml | board-mapping.xml | BoardDAO.java | SqlSessionFactoryBean.java | BoardServiceClient.java

```
1  package kr.ac.inje.comsi.board.impl;
2
3  import java.util.List;
4
5  import org.apache.ibatis.session.SqlSession;
6
7  import kr.ac.inje.comsi.board.BoardVO;
8  import kr.ac.inje.comsi.board.SqlSessionFactoryBean;
9
10 public class BoardDAO {
11
12     private SqlSession mybatis;
13
14     public BoardDAO(){
15         mybatis = SqlSessionFactoryBean.getSqlSessionInstance();
16     }
17
18     public void insertBoard(BoardVO vo) {
19         mybatis.insert("BoardDAO.insertBoard", vo);
20         mybatis.commit();
21     }
22
23
24     public void updateBoard(BoardVO vo) {
25         mybatis.update("BoardDAO.updateBoard", vo);
26         mybatis.commit();
27     }
28
```

이어서…

```
29      public void deleteBoard(BoardVO vo) {
30          mybatis.delete("BoardDAO.deleteBoard", vo);
31          mybatis.commit();
32      }
33
34
35      public BoardVO getBoard(BoardVO vo) {
36          return (BoardVO) mybatis.selectOne("BoardDAO.getBoard", vo);
37      }
38
39      public List<BoardVO> getBoardList(BoardVO vo) {
40          return mybatis.selectList("BoardDAO.getBoardList", vo);
41      }
42
43  }
44
```

테스트 클라이언트-BoardServiceClient 클래스 수정

root-context.xml | mybatis-config.xml | board-mapping.xml | BoardDAO.java | SqlSessionFactoryBean.java | BoardServiceClient.java

```java
package kr.ac.inje.comsi.board;

import java.util.List;

import kr.ac.inje.comsi.board.impl.BoardDAO;

public class BoardServiceClient {

    public static void main(String[] args) {
        BoardDAO boardDAO = new BoardDAO();

        BoardVO vo = new BoardVO();
        vo.setTitle("myBatis 제목");
        vo.setWriter("김길동");
        vo.setContent("myBatis 내용입니다....");
        //boardDAO.insertBoard(vo);

        List<BoardVO> boardList = boardDAO.getBoardList(vo);
        for (BoardVO board:boardList) {
            System.out.println("-->" + board.toString());
        }
    }

}
```

Console | Progress | Problems | Markers

```
<terminated> BoardServiceClient [Java Application] C:₩Program Files₩Java₩jre1.8.0_171₩bin₩javaw.exe (2018. 11. 12. 오후 10:16:53)
-->BoardVO [seq=5, title=글 등록, writer=글 등록, content=글 수정테스트, regDate=2018-09-10, cnt=0]
-->BoardVO [seq=4, title=JDBD 테스트2, writer=관리자, content=JDBC 테스트2....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=3, title=JDBD 테스트, writer=관리자, content=JDBC 테스트....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=2, title=임시 제목, writer=홍길동, content=임시 내용 ....., regDate=2018-05-12, cnt=0]
-->BoardVO [seq=1, title=가입인사, writer=관리자, content=잘 부탁드립니다...., regDate=2018-04-08, cnt=0]
```

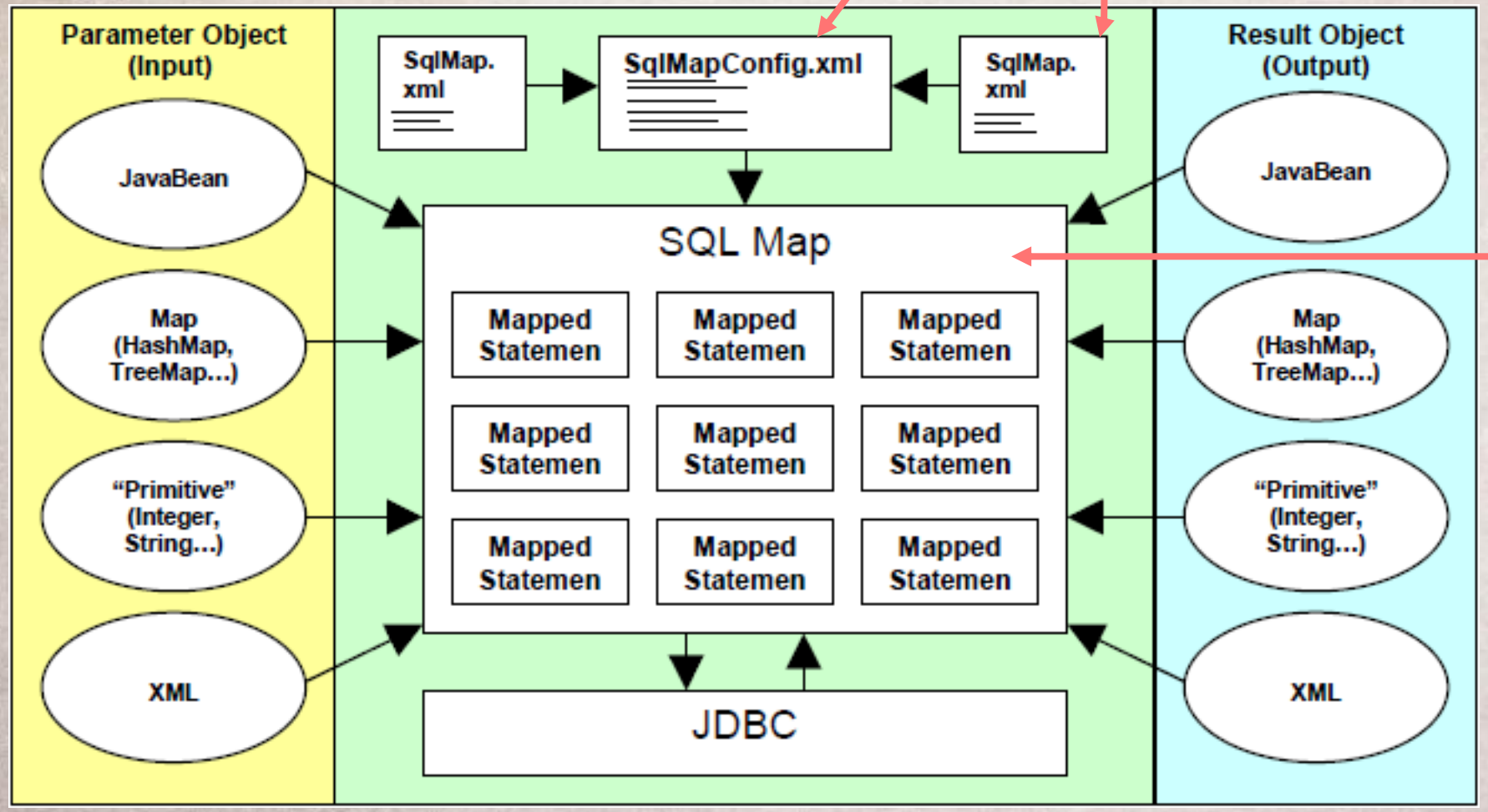
02. Mapper XML 파일 설정

2.1.1 Mybatis 프레임워크 구조

Mybatis 메인 환경 설정 파일
(mybatis-config.xml)

SQL Mapper파일
(board-mapping.xml)

입력



실행할 SQL명령어가
Map 구조로 저장관리

출력

http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:rte2:psl:data_access

2.1.2 SQL Mapper 파일 구조

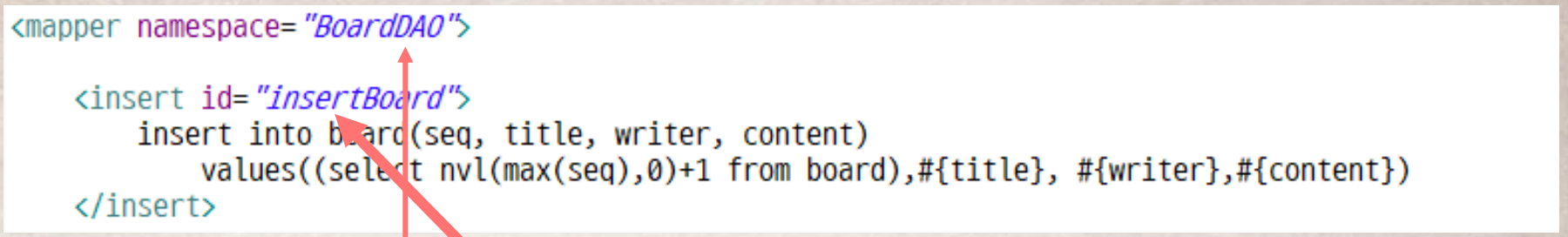
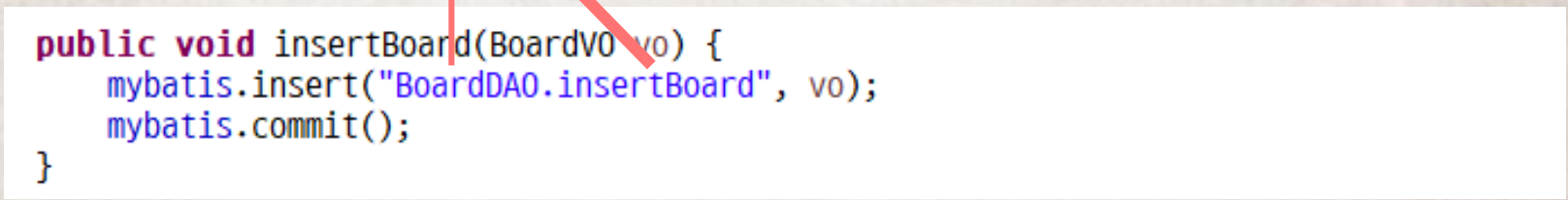
```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
                        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="BoardDAO">

    <insert id="insertBoard">
        insert into board(seq, title, writer, content)
            values((select nvl(max(seq),0)+1 from board),#{title}, #{writer},#{content})
    </insert>

    <select id="getBoard" resultType="board">
        select * from board where seq=#{seq}
    </select>

    <select id="getBoardList" resultType="board">
        select * from board order by seq desc
    </select>

    <delete id="deleteBoard">
        delete board where seq=#{seq}
    </delete>

    <update id="updateBoard">
        update board set title=#{title}, content=#{content} where seq=#{seq}
    </update>

</mapper>
```

DTD 선언 부분

mapper 루트 엘리먼트

SQL 매핑 부분

Mapper 파일과 DAO 클래스 연결

Mapper XML	
DAO 클래스	

- Mapper 파일에 SQL 명령어를 등록할 때는 SQL 구문의 종류에 따라 적절한 엘리먼트를 사용한다.
- INSERT 구문은 <insert>, SELECT구문은 <select> 엘리먼트를 사용
- 각 엘리먼트에서 사용할 수 있는 속성들이 다르므로 그 용도와 의미를 이해해야 한다.

2.1.3 <select> 엘리먼트

Mapper XML

```
<select id="getBoard" parameterType="board" resultType="board">
    select * from board where seq=#{seq}
</select>

<select id="getBoardList" resultType="board">
    select * from board order by seq desc
</select>
```

- id – 필수 속성으로 Mapper 파일들 내에서 유일해야하며 DAO에서 호출가능하다.
- parameterType – SQL 실행에 필요한 데이터를 외부로부터 받아야 할 때 사용하며 일반적으로 기본형이나 VO형태의 클래스를 지정
- resultType – SQL 구문이 실행된 ResultSet 결과를 매핑할 객체 지정

2.1.4 기타 엘리먼트

<insert> 엘리먼트	<insert id="insertBoard" parameterType="board"> insert into board(seq, title, writer, content) values((select nvl(max(seq),0)+1 from board),#{title}, #{writer},#{content}) </insert>
<update> 엘리먼트	<update id="updateBoard" parameterType="board"> update board set title=#{title}, content=#{content} where seq=#{seq} </update>
<delete> 엘리먼트	<delete id="deleteBoard" parameterType="board"> delete board where seq=#{seq} </delete>

2.2 SQL Mapper XML 추가 설정

- 2.2.1 resultMap 속성 설정
  - 일반적으로 resultType 속성으로 SQL 실행결과를 넘겨주는 방식을 사용한다.
    - 하나의 테이블 결과를 넘겨줄 때 테이블의 VO객체 사용
  - 2개 테이블을 join하여 넘겨줄 때는 적용할 수 없다. → resultMap 사용
- 2.2.2 CDATA Section 사용

```
<select id="getBoard" parameterType="board" resultType="board">
    select * from board where seq=#{seq}
</select>
```

  - CDATA(Character DATA)는 XML 파서에 의해서 해석되지 않는다. 즉, CDATA 안의 내용이 그대로 DB에 전달되어 실행된다.

```
<select id="getBoard" parameterType="board" resultType="board">
    <![CDATA[
        select * from board where seq=#{seq} < 5
    ]]>
</select>
```

2.3 Mybatis JAVA API

- 2.3.1 SqlSessionFactoryBuilder 클래스
  - Mybatis로 DAO 클래스의 CRUD메소드를 구현하려면 Mybatis에서 제공하는 SqlSession객체를 사용해야 함.
  - SqlSession객체는 SqlSessionFactory로부터 얻어야한다.
  - 따라서, 가장 먼저 할 작업은 SqlSessionFactory 객체를 생성하는 일임.
  - SqlSessionFactory는 SqlSessionFactoryBuild의 build()메소드를 이용
    - Mybatis 설정 파일()을 로딩하여 SqlSessionFactory객체를 생성

```
Reader reader = Resources.getResourceAsReader("mybatis-config.xml");
sessionFactory = new SqlSessionFactoryBuilder().build(reader);
```

- 2.3.2 SqlSessionFactory 클래스
  - SqlSession 객체에 대한 생성 역할을 수행
  - SqlSessionFactory객체는 openSession()이라는 메소드를 통해 SqlSession 객체를 획득

```
sessionFactory.openSession();
```

• 2.3.3 유틸리티 클래스 작성 – 싱글톤 디자인 패턴으로 작성

root-context.xml | mybatis-config.xml | board-mapping.xml | BoardDAO.java | SqlSessionFactoryBean.java | BoardServiceClient.java

```java
package kr.ac.inje.comsi.board;

import java.io.Reader;

import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

public class SqlSessionFactoryBean {
    private static SqlSessionFactory sessionFactory = null;

    static {
        try {
            if (sessionFactory == null) {
                Reader reader = Resources.getResourceAsReader("mybatis-config.xml");
                sessionFactory = new SqlSessionFactoryBuilder().build(reader);
            }
        }catch(Exception e) {
            e.printStackTrace();
        }
    }

    public static SqlSession getSqlSessionInstance() {
        return sessionFactory.openSession();
    }
}
```

2.3.4 SqlSession 객체

(1) selectOne() : 오직 하나의 데이터를 검색하는 SQL구문 실행할 때
- public Object selectOne(String statement)
- public Object selectOne(String statement, Object parameter)
- statement매개변수는 Mapper XML 파일에 등록된 SQL의 아이디(ID)

(2) selectList() : 여러 개의 데이터가 검색되는 SQL구문을 실행할 때
- public Object selectList(String statement)
- public Object selectList(String statement, Object parameter)

(3) insert(), update(), delete() : 각각 INSERT, UPDATE, DELETE SQL구문을 실행할 때
- public int insert(String statement, Object parameter)
- public int update(String statement, Object parameter) throws SQLException
- public int delete(String statement, Object parameter) throws SQLException

➔ 사용 방법은 BoardDAO 클래스에서 확인